Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

TRABAJO FINAL DE GRADO

Título: Simulación y análisis de sistemas de distribución de clave cuántica

Titulación: Grado en Ingeniería de Sistemas de Telecomunicación

Autor: Albert Valls Marin

Directores: Pere Bruna Escuer y Santiago Torres Gil

Fecha: 8 de febrero del 2018

Resumen

El objetivo de este proyecto es analizar el comportamiento de tres protocolos de distribución de clave cuántica (*QKD*): BB84, B92 y EPR, con la intención de comparar, para diferentes escenarios, el funcionamiento de cada uno de ellos.

En primer lugar se definen los conceptos elementales de la computación cuántica: superposición, qubit, colapso de estados y entrelazamiento. A continuación, se realiza una breve introducción de los fundamentos de la criptografía cuántica.

Tras introducir el marco teórico, se presenta el desarrollo, mediante *MATLAB*, del *script* que permitirá simular la sesión *QKD* para cada protocolo.

Para el conjunto de escenarios estudiados, se analizan las simulaciones obtenidas con el objetivo de comparar el funcionamiento de cada protocolo. De esta forma, podemos analizar la robustez de los protocolos en función de los diferentes factores que pueden actuar durante una sesión *QKD* como el error introducido por el canal, tales como el error de medida en la recepción y, desde un punto de vista de seguridad, el que más nos interesa detectar es la medida realizada por un intruso.

Por último, se analiza un escenario real y se estiman las pérdidas en los canales cuánticos siguientes: la fibra óptica y el espacio libre.

Title: Simulation and analysis of quantum key distribution systems

Titulation: Degree in Telecommunication Systems Engineering

Author: Albert Valls Marin

Directors: Pere Bruna Escuer and Santiago Torres Gil

Date: February 8 th 2018

Overview

The goal of this project is analyze the behavior of three quantum key distribution protocols (*QKD*): BB84, B92 and EPR, with the intention of comparing each of them for different scenarios.

First of all, the elementary concepts of quantum computing are defined: superposition, qubit, state collapse and entanglement. Next, a brief introduction of the basics of quantum cryptography is made.

After introducing the theoretical frame, the development through *MATLAB* of the script that will simulate the *QKD* session for each protocol is presented.

For the set of scenarios studied, the simulations obtained are analyzed in order to compare the operation of each protocol. In this way, we can analyze the robustness of the protocols according to the different factors that can act during a *QKD* session such as the error introduced by the channel, the measurement error in the reception and, from a security point of view, the one we are most interested in detecting is the measurement made by an intruder.

Finally, a real scenario is analyzed and losses for the following quantum channels are estimated: optical fiber and free space.

Agradecimientos

Me gustaría agradecer a Pere Bruna y Santiago Torres, directores de este trabajo, su valiosa ayuda y motivación desde que cursé la asignatura *Tecnologías de Información Cuántica* hasta hoy, día que finalizo mi trabajo final de grado. Definitivamente me habéis brindado todas las herramientas para completar mi proyecto satisfactoriamente.

ÍNDICE

INTRODUCCIÓN

La distribución de clave cuántica (*QKD*) permite alcanzar nuevas propiedades de seguridad, únicamente limitadas por las leyes de la mecánica cuántica. Esta seguridad se basa en la detección inevitable del rastro en la intervención de quién realiza la escucha (*eavesdropper*). El emisor (Alice) y receptor (Bob) autentificado para el intercambio, pueden decidir si descartar la información corrupta y generar una nueva sesión *QKD*. Desde su aparición en los años ochenta como concepto teórico prometedor, una gran variedad de protocolos han surgido y se han demostrado en muchos escenarios del mundo real.

La división de este trabajo está comprendida en cuatro capítulos donde se realiza un estudio teórico y práctico de los protocolos BB84, B92 y EPR. El objetivo principal de este trabajo es comprobar el comportamiento de cada uno de ellos para diferentes escenarios: error de canal cuántico, error de medida y participación de un intruso (Eva). Para ello, se ha realizado, mediante *MATLAB*, un *script* que permita simular una sesión *QKD* para cada uno de los tres protocolos, con un número de parámetros de entrada que permitan simular un escenario realista. Por último, se analiza la diferencia entre la fibra óptica y el espacio libre como canales cuánticos utilizados.

En el primer capítulo, se establecen los conceptos básicos de la mecánica cuántica necesarios para comprender el funcionamiento de la criptografía cuántica, a la vez que se realiza una comparativa respecto a la mecánica clásica. Seguidamente, en el segundo capítulo, se trata de forma teórica el funcionamiento de los tres protocolos *QKD*, acompañados de un ejemplo práctico con la intención de aclarar las diferentes acciones realizadas por los participantes durante la comunicación. A continuación, en el tercer capitulo, se realiza la explicación de los diferentes parámetros de entrada y salida del *script* realizado en *MATLAB* para dar paso al conjunto de simulaciones ejecutadas para cada escenario. Por último, en el cuarto capítulo, se estudia el comportamiento de las leyes de la mecánica cuántica que afectan a la transmisión de información (estados cuánticos) y se analiza el comportamiento de los enlaces de fibra óptica y espacio libre.

El uso de *MATLAB* viene dado por su versatilidad, por la formación recibida durante el grado universitario en las diferentes funcionalidades del software, la fácil disposición de una licencia de uso académico ofrecida por la universidad y por la oportunidad de compartir el *script* para ayudar a los alumnos interesados en comprender y profundizar en la materia tratada en esta memoria.

CAPÍTULO 1. MECÁNICA CUÁNTICA Y CRIPTOGRAFÍA CUÁNTICA

En este primer capítulo, se introduce de forma breve los fundamentos de la mecánica cuántica con el objetivo de tratar el funcionamiento de la criptografía cuántica.

1.1. Mecánica cuántica: conceptos básicos

A lo largo del proyecto, se hace mención a un conjunto de conceptos que es necesario definir (ver [7]). Para ello, se ha creído conveniente presentar cada uno de ellos de manera individual:

1. Superposición: Una partícula cuántica (fotones, electrones…) puede hallarse en más de un estado a la vez, es decir, pueden coexistir simultáneamente en todas sus posibles configuraciones físicas.

2. Qubit: Unidad básica de medición de la información cuántica. A diferencia del bit clásico, que sólo existe en uno de los dos estados, 0 o 1, el bit cuántico, además de los estados 0 y 1 que se presentan con los símbolos $|0>$ y $|1>$, pueden presentar superposición de estados, es decir, un qubit $|q>$ cualquiera se puede escribir como:

$$|q> = \propto |0> + \beta |1> \qquad \textbf{(1.1)}$$

   Donde α y β, denominados amplitudes, pueden ser números complejos cumpliendo la condición de normalización:

$$|\alpha|^2 + |\beta|^2 = 1 \qquad \textbf{(1.2)}$$

   Así pues, un qubit se puede hallar en ambos estados $|0> y |1>$ a la vez previamente a la medida.

3. Colapso de estados: La acción de medir una partícula provoca que se obtenga un resultado de sólo una de las posibles configuraciones, alterando su estado superpuesto. El colapso en uno u otro estado es un fenómeno probabilístico que viene determinado por las amplitudes α y β, siendo $P(|0>) = \alpha^2$ y $P(|1>) = \beta^2$, las probabilidades de colapsar en los estados $|0>$ y $|1>$, respectivamente.

4. Entrelazamiento: Un par de partículas cuánticas pueden tener estados correlados, esto provoca que la medición sobre una de ellas determina el estado de la otra.

1.2. Fundamentos de la criptografía cuántica

La criptografía es una técnica empleada para enviar mensajes cifrados. Su metodología no fue analizada hasta el siglo XX, aunque hay que resaltar, que su uso se remonta a siglos anteriores, dado que a lo largo de la historia ha persistido la necesidad de enviar información confidencial y por ende, cifrada.

A día de hoy, se pueden diferenciar dos métodos criptográficos (ver [10]), teniendo ambos el mismo fin: asegurarse de que el texto cifrado, mediante un conjunto de claves, no se pueda descifrar. En primer lugar, se encuentran los protocolos de clave pública, caracterizados por el uso de dos claves para cifrar y descrifrar el mensaje. De manera más detallada, se emplea una clave pública para el cifrado y una clave privada para el descifrado de la información. Destacar, que en este caso los receptores pueden ser múltiples. En relación al segundo método criptográfico, los protocolos de clave privada o secreta utilizan una sola clave compartida entre el emisor y el receptor, a diferencia del primero, el numero de receptores del mensaje es más reducido, en concreto dos, por ejemplo, Alice y Bob (convenio universal).

La disciplina del criptoanálisis juega un papel muy importante en el ámbito de la criptografía, ya que tiene como propósito principal descodificar la información cifrada. Ante este escenario, la necesidad continuada de desarrollar métodos de cifrado más seguros y complejos, con el fin asegurar la confidencialidad de la información transmitida, es un gran desafío para los criptógrafos. Como resultado, la criptografía cuántica es un claro ejemplo, dado que con el surgimiento de esta disciplina, se han llevado a cabo métodos de cifrado más seguros, fundamentados por la mecánica cuántica.

La criptografía cuántica se propuso en el año 1970, pero no es hasta el 1984 cuando se publica el primer protocolo (ver [5]). Las actuales técnicas permiten generar de forma segura la clave secreta, aún si un intruso realiza una medición durante la comunicación. Esto es consecuencia del principio de incertidumbre de Heisenberg, que nos dice que el proceso de medir en un sistema cuántico perturba dicho sistema.

CAPÍTULO 2. DISTRIBUCIÓN DE CLAVE CUÁNTICA

La distribución de clave cuántica es una técnica utilizada en el contexto de la criptografía cuántica para generar una clave perfectamente aleatoria compartida por un emisor y un receptor (generalmente conocidos como Alice y Bob), al tiempo que se asegura que nadie más tenga la oportunidad de aprender acerca de la clave, interceptando el canal de comunicación utilizado durante el proceso (generalmente conocido como man in the middle o Eva).

Las técnicas actuales de distribución de clave cuántica (*QKD*) utilizan protocolos que mediante técnicas de reconciliación de información y ampliación de privacidad permiten alcanzar un alto nivel de seguridad. Para ello, existen unas premisas mínimas necesarias, como que las leyes de la física cuántica son correctas y que Alice y Bob son capaces de autentificarse mutuamente (Eva no debería de ser capaz de hacerse pasar por Alice o Bob).

En este capítulo, se realizará el estudio de tres de los protocolos de distribución de clave cuántica más importantes y que presentan un conjunto de características distintas entre ellos, estos son: BB84, B92 y EPR (también conocido como E91).

2.1. Protocolo BB84

El protocolo BB84 fue descrito y publicado por C. H. Bennet y G. Brassard en el año 1984 (ver [1]). Se trata de un protocolo asimétrico, de manera que es Alice quien genera la sesión de distribución de clave cuántica.

Alice utiliza un generador de bits aleatorios para producir dos secuencias $A_1 = \{a_1, a_2, \ldots, a_N\}$ y $A_2 = \{a'_1, a'_2, \ldots, a'_N\}$ cada una de longitud N. El objetivo es producir una secuencia $\left|\psi\right\rangle$ de N-qubits, que podemos expresar de la siguiente manera:

$$\left|\psi\right\rangle = \left|\psi_{a_1 a'_1}\right\rangle \otimes \left|\psi_{a_2 a'_2}\right\rangle \ldots \otimes \left|\psi_{a_N a'_N}\right\rangle = \bigotimes_{k=1}^{N} \left|\psi_{a_k a'_k}\right\rangle \qquad \textbf{(2.1)}$$

Utiliza la base canónica con fotones polarizados horizontalmente para representar un $\left|0\right\rangle$ y verticalmente para representar un $\left|1\right\rangle$. Pero también

utiliza una base transversal, utilizando una polarización 45º para representar un $|+\rangle$ y de 135º para representar un $|-\rangle$ (ver [8]).

En función del valor de a_k y a'_k, Alice genera un qubit preparado en un estado cuántico que pertenece a una de las dos bases conjugadas siguientes: $Z = \{|0\rangle, |1\rangle\}$, $X = \{|+\rangle, |-\rangle\}$. De acuerdo al siguiente convenio mostrado en la tabla 2.1.

Tabla 2.1. Convenio utilizado para generar el estado cuántico del emisor

a_k	a'_k	$\|\psi_{a_k a'_k}\rangle$	Qubit
0	0	$\|\psi_{00}\rangle$	$\|0\rangle$
0	1	$\|\psi_{01}\rangle$	$\|1\rangle$
1	0	$\|\psi_{10}\rangle$	$\|+\rangle$
1	1	$\|\psi_{11}\rangle$	$\|-\rangle$

Bob recibe el estado cuántico creado por Alice y utiliza un generador de bits aleatorio para producir una secuencia $B = \{b_1, b_2, \ldots, b_N\}$ de longitud N. La base que utilizará en sus medidas será la Z o la X según el siguiente convenio mostrado en la tabla 2.2.

Tabla 2.2. Convenio utilizado para generar las bases de medida del receptor

Bit	Base
0	Z
1	X

Bob realiza la medida de todos los qubits según esta elección de bases, teniendo las siguientes posibilidades:

1. Mide con Z el qubit $|0\rangle$ y obtiene el bit 0 con un 100% de probabilidad.
2. Mide con Z el qubit $|1\rangle$ y obtiene el bit 1 con un 100% de probabilidad.
3. Mide con Z el qubit $|+\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
4. Mide con Z el qubit $|-\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.

5. Mide con X el qubit $|0\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
6. Mide con X el qubit $|1\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
7. Mide con X el qubit $|+\rangle$ y obtiene el bit 0 con un 100% de probabilidad.
8. Mide con X el qubit $|-\rangle$ y obtiene el bit 1 con un 100% de probabilidad.

En resumen, la base Z mide $|0\rangle$ o $|1\rangle$ mientras que la base X mide $|+\rangle$ o $|-\rangle$, por tanto, con un 100% de probabilidad cuando se utiliza la base correcta respecto al estado. En caso contrario, la probabilidad es del 50%. Esto es debido a que el estado $|+\rangle$ y $|-\rangle$ son superposiciones del $|0\rangle$ y el $|1\rangle$ y viceversa. En cualquier caso, en promedio hay un 50% de probabilidades de que se realice una medición “correcta” o “incorrecta”, es decir, con estados que pueden o no corresponder a los de Alice.

Usando un canal público (posiblemente interceptable), Alice anuncia a Bob la secuencia de bases de medición de estados $A_1 = \{a_1, a_2, \ldots, a_N\}$ y Bob responde con las posiciones donde $a_k = b_k$. Si el número de posiciones enviadas por Bob es menor que dos, la sesión del protocolo debe abortarse y reiniciarse con nuevas secuencias aleatorias. Es importante destacar, que en ningún momento existe la difusión de las direcciones de las polarizaciones elegidas. Una vez realizado el intercambio de información (vía telefónica, internet…), ambos descartan todos los fotones con una base “incorrecta”. Los restantes, representan una nueva secuencia de bits (*reconciled key*) que tendrían que ser idénticos para Alice y Bob y deberían ser conocidos sólo por ellos, siempre que la comunicación no haya sido manipulada por nadie.

Para obtener el número de bits perdidos es necesario recurrir a técnicas de ampliación de privacidad y generar una nueva clave (*secret key*) a partir de la *reconciled key*. Para realizar esta operación existen numerosos algoritmos al efecto en base a la utilización de las funciones universales de troceado (ver [9]). Dado que la aplicación de privacidad no afecta al estudio de los protocolos que realizaremos, en la presente memoria se ha optado por utilizar el operador lógico XOR para ejemplificar de forma sencilla este proceso, cuya tabla de la verdad es la indicada en la tabla 2.3.

Tabla 2.3. Tabla de la verdad del operador lógico XOR ($\oplus$)

A	B	A $\oplus$ B
0	0	0
0	1	1
1	0	1
1	1	0

La longitud de la *secret key* es la misma que la secuencia aleatoria inicial, es decir, N.

Podemos concluir de manera anticipada que el algoritmo anterior es suficiente para garantizar la estricta seguridad de la comunicación. De hecho, todavía hay posibilidades de que el canal público utilizado por Alice y Bob esté corrompido por los efectos de ruido aleatorio y/o también por las acciones de manipulación de Eva.

Eva tendría que detectar las polarizaciones de los fotones sin conocer las bases correspondientes. En los casos en que la suposición de Eva con respecto a las bases sea incorrecta, obtendrá resultados aleatorios. Si Eva envía fotones con estas polarizaciones, los resultados de Bob también serán aleatorios en los casos en que la suposición de Bob sea correcta. La mecánica cuántica no permitiría a Eva realizar una medición de las polarizaciones sin alterar los estados de los fotones.

Independientemente de que esto ocurra, Alice y Bob pueden probar comparando cierto número de bits obtenidos a través del canal de información público. Si estos bits coinciden, saben que los otros también son correctos y finalmente pueden usarse para la transmisión de datos reales.

Previamente a la realización de un ejemplo práctico, es necesario definir el concepto de tasa de error binario (BER) y eficiencia (Eff).

Entendemos como tasa de error binario (BER) el cociente entre número de bits erróneos n_{error} (número de bits dispares entre las claves de Alice y Bob) y el número total de bits generados inicialmente n_{bits}:

$$BER = \frac{n_{error}}{n_{bits}} \tag{2.2}$$

En cuanto a la eficiencia (Eff), en el numerador se encuentra la diferencia entre el número de bits de la *reconciled key* n_{rk} y el número de bits erróneos n_{error} y, en el denominador, el número total de bits generados inicialmente n_{bits}:

$$Eff = \frac{n_{rk} - n_{error}}{n_{bits}} \tag{2.3}$$

2.1.1. Ejemplo de sesión de distribución de clave cuántica

El protocolo BB84 empieza con Alice generando dos secuencias aleatorias de bits clásicos (cbits). Los cbits de estas secuencias los llamamos a_k y a'_k. Podrían ser, respectivamente, los siguientes:

$$10110111$$
$$01010101$$

En función del valor de a_k y a'_k, Alice crea un fotón (por tanto, un qubit) $\left|0\right\rangle$, $\left|1\right\rangle$, $\left|+\right\rangle$ o $\left|-\right\rangle$ (Tabla 2.1). Por tanto, en este ejemplo, el estado cuántico que tendrá Alice y que enviará a Bob sería el siguiente:

$$\left|+\right\rangle \left|1\right\rangle \left|+\right\rangle \left|-\right\rangle \left|0\right\rangle \left|-\right\rangle \left|+\right\rangle \left|-\right\rangle$$

Bob recibe este estado cuántico y ha de decidir en que base mide cada qubit. Para ello, genera otra secuencia aleatoria de cbits que llamamos b_k. Por ejemplo:

$$10111000$$

La base que utilizará será la Z o la X (Tabla 2.2). Por tanto, con esta secuencia en particular, las bases de medida serán:

$$XZXXXZZZ$$

Ahora realiza la medida de todos los qubits según esta elección de bases, obtendrá los cbit asociados a las bases Z y X obteniendo la secuencia que llamaremos b'_k. De esta manera, la secuencia b'_k que obtenemos es la siguiente:

$$0101\frac{0}{1}\frac{0}{1}\frac{0}{1}\frac{0}{1}$$

Donde 0/1 indica que podemos tener un 0 o un 1 con una probabilidad del 50% cada uno. Por tanto, una posible secuencia sería la siguiente:

$$01011011$$

Alice envía la secuencia a_k a Bob por un canal público y Bob responde con las posiciones donde $a_k = b_k$. De esta manera, las posiciones serían:

$$1^a,\ 2^a,\ 3^a \text{ y } 4^a$$

Alice se quedará con aquellos cbits de la secuencia a'_k que correspondan a las posiciones que Bob ha enviado, generando una secuencia (*reconciled key*) de cbits que llamamos a''_k:

$$A_{reconciled-key} = 0101$$

Mientras que Bob, a partir de los cbits de la secuencia b'_k que corresponden a las posiciones donde $a_k = b_k$, genera su propia *reconciled key* de cbits que llamamos b''_k:

$$B_{reconciled-key} = 0101$$

A continuación, Alice y Bob generan una nueva clave (*secret key*) con un mayor nivel de seguridad (ampliación de privacidad) a partir de la *reconciled key*. Para realizar esta operación, se utiliza tal y como hemos comentado anteriormente el operador lógico XOR ($\oplus$). En este ejemplo, Alice y Bob generan la siguiente *secret key* de 8 cbits de longitud:

$$a''_5 = a''_1 \oplus a''_2 = 0 \oplus 1 = 1 \qquad b''_5 = b''_1 \oplus b'_2 = 0 \oplus 1 = 1$$
$$a''_6 = a''_2 \oplus a''_3 = 1 \oplus 0 = 1 \qquad b''_6 = b''_2 \oplus b''_3 = 1 \oplus 0 = 1$$
$$a''_7 = a''_3 \oplus a''_4 = 0 \oplus 1 = 1 \qquad b''_7 = b''_3 \oplus b''_4 = 0 \oplus 1 = 1$$
$$a''_8 = a''_4 \oplus a''_5 = 1 \oplus 1 = 0 \qquad b''_8 = b''_4 \oplus b''_5 = 1 \oplus 1 = 0$$

$$A_{secret-key} = 01011110$$
$$B_{secret-key} = 01011110$$

Como se trata de un ejemplo teórico con condiciones ideales, tanto Alice como Bob tienen la misma clave y, por tanto, la BER es del 0%.

Resumiendo todos los pasos en formato tabla donde la zona de color corresponde a la parte de Alice y la blanca a la parte de Bob (en rojo se indica todos los números que son resultado de un proceso aleatorio) se concluye el resultado mostrado en la tabla 2.4.

Tabla 2.4. Resumen del desarrollo práctico para el protocolo BB84

a_k	1	0	1	1	0	1	1	1
a'_k	0	1	0	1	0	1	0	1
Qubit	$\lvert+\rangle$	$\lvert 1\rangle$	$\lvert+\rangle$	$\lvert-\rangle$	$\lvert 0\rangle$	$\lvert-\rangle$	$\lvert+\rangle$	$\lvert-\rangle$
b_k	1	0	1	1	1	0	0	0
Base	X	Z	X	X	X	Z	Z	Z
Medida	$+1$	-1	$+1$	-1	± 1	± 1	± 1	± 1
Posibles b'_k	0	1	0	1	0/1	0/1	0/1	0/1
b'_k	0	1	0	1	1	0	1	1
$A_{secret-key}$	0	1	0	1	1	1	1	0
$B_{secret-key}$	0	1	0	1	1	1	1	0

Por último, obtenemos la eficiencia (Eff) para este ejemplo utilizando la ecuación descrita en (2.3):

$$Eff = \frac{n_{rk} - n_{error}}{n_{bits}} = \frac{4}{8} = 0.5$$

Un 50% de los qubits generados inicialmente son utilizados para elaborar la *reconciled key* de esta comunicación.

2.2. Protocolo B92

El protocolo B92, propuesto por C. H. Bennet el año 1992 (ver [1]), es una versión simplificada del protocolo BB84, en la que únicamente utilizan dos estados no ortogonales. Se trata de un protocolo asimétrico, de manera que es Alice quien genera la sesión de distribución de clave cuántica.

Alice utiliza un generador de bits aleatorios para producir una secuencia $A = \{a_1, a_2, \ldots, a_N\}$ de longitud N.

En función del valor de a_k, Alice genera un qubit de acuerdo al siguiente convenio mostrado en la tabla 2.5.

Tabla 2.5. Convenio utilizado para generar el estado cuántico del emisor

a_k	$\left\|\psi_{a_k}\right\rangle$	Qubit
0	$\left\|\psi_0\right\rangle$	$\left\|0\right\rangle$
1	$\left\|\psi_1\right\rangle$	$\left\|+\right\rangle$

A continuación, produce una secuencia $\left|\psi\right\rangle$ de N-qubits, que podemos expresar de la siguiente manera:

$$\left|\psi\right\rangle = \left|\psi_{a_1}\right\rangle \otimes \left|\psi_{a_2}\right\rangle \ldots \otimes \left|\psi_{a_N}\right\rangle = \bigotimes_{k=1}^{N} \left|\psi_{a_k}\right\rangle \qquad \textbf{(2.4)}$$

Bob recibe el estado cuántico creado por Alice y utiliza un generador de bits aleatorio para producir una secuencia $B = \{b_1, b_2, \ldots, b_N\}$ de longitud N. La base que utilizará será la Z o la X (Tabla 2.2).

Ahora realiza la medida de todos los qubits según esta elección de bases, teniendo las siguientes posibilidades:

1. Mide con Z el qubit $\left|0\right\rangle$ y obtiene el bit 0 con un 100% de probabilidad.
2. Mide con Z el qubit $\left|+\right\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
3. Mide con X el qubit $\left|0\right\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
4. Mide con X el qubit $\left|+\right\rangle$ y asigna el bit 0 con un 100% de probabilidad.

Cada uno de los estados que mide Bob se asocian a un bit creando la secuencia b'_k.

Usando un canal público (posiblemente interceptable), Bob anuncia a Alice las posiciones de la secuencia b'_k $(0101...)$ donde $b'_k = 1$. Esto es debido a que, a diferencia del BB84, sólo cuando Bob obtiene un 1 tras la medida, puede estar seguro que las bases utilizadas son opuestas con respecto a Alice.

Bob genera una secuencia post medida b'_k con un 50% de probabilidad de generar un 0 y otro 50% de generar un 0/1 (25% de generar un 1). Por tanto, únicamente un 25% de los bits serán utilizados para generar la *reconciled key* (bits útiles). Esto implica que la eficiencia del protocolo sea del 25%, es decir, un 50% menor que para el protocolo BB84.

Una vez realizado el intercambio de información (vía telefónica, internet…), ambos descartan todos los fotones con una base “incorrecta”. Bob utiliza la operación lógica NOT con los restantes y al igual que Alice, mantienen una nueva secuencia de bits (*reconciled key*) que tendrían que ser idénticos para Alice y Bob y deberían ser conocidos solo por ellos, siempre que la comunicación no haya sido manipulada por nadie.

El resto del protocolo B92 es igual que el BB84, es decir, se aplican técnicas de ampliación de seguridad (*secret key*) y en el caso de que el canal público utilizado por Alice y Bob esté corrompido (efectos de ruido aleatorio y/o también por las acciones de manipulación de Eva), pueden probar comparando cierto número de bits y verificar si estos coinciden o no.

Para ver un ejemplo de sesión de distribución de clave cuántica para el protocolo B92 consultar el anexo A.1.

2.3. Protocolo EPR

El protocolo EPR, también conocido como E91, fue propuesto por A. K. Eckert el año 1991 (ver [1]). Se trata, a diferencia de los anteriores, de un protocolo simétrico, de manera que, tanto Alice como Bob pueden generan una sesión de distribución de clave cuántica.

Este protocolo basa su seguridad en el uso de pares EPR (estado cuántico de 2 qubits entrelazados). Alice y Bob comparten, a través de un canal cuántico, un conjunto de pares EPR, basados en uno de los cuatro estado de Bell:

$$\left|\beta_{00}\right\rangle = \frac{1}{\sqrt{2}}(\left|00\right\rangle + \left|11\right\rangle)$$

$$\left|\beta_{01}\right\rangle = \frac{1}{\sqrt{2}}(\left|01\right\rangle + \left|10\right\rangle)$$

$$\left|\beta_{10}\right\rangle = \frac{1}{\sqrt{2}}(\left|00\right\rangle + \left|11\right\rangle)$$

$$\left|\beta_{11}\right\rangle = \frac{1}{\sqrt{2}}(\left|01\right\rangle + \left|10\right\rangle)$$

En este caso se utilizará, por ejemplo, el estado de Bell $\left|\beta_{00}\right\rangle$ (convenio universal).

El estado cuántico del sistema de Alice y Bob esta definido por el tensor N-EPR, donde N es el número de pares EPR, que podemos expresar de la siguiente manera:

$$\left|\psi\right\rangle = \left|\beta_{00}\right\rangle_1 \otimes \left|\beta_{00}\right\rangle_2 \ldots \otimes \left|\beta_{00}\right\rangle_N = \bigotimes_{k=1}^{N} \left|\beta_{00}\right\rangle_k \qquad \textbf{(2.5)}$$

Como en los protocolos anteriores, Alice y Bob utilizan un generador de bits aleatorios para producir dos secuencias $A_1 = \{a_1, a_2, \ldots, a_N\}$ y $B_1 = \{b_1, b_2, \ldots, b_N\}$ cada una de longitud N. Los valores del bit a_k y b_k determinan su elección de base de medida (Tabla 2.2) según corresponda al primer (Alice) y segundo (Bob) qubit, respectivamente, para cualquiera de los pares EPR del tensor N-EPR. Los resultados de las mediciones de Alice y Bob generan las secuencias de bits clásicos aleatorias $A_2 = \{a'_1, a'_2, \ldots, a'_N\}$ y $B_2 = \{b'_1, b'_2, \ldots, b'_N\}$, respectivamente.

Se considera que el estado entrelazado $\left|\beta_{00}\right\rangle$ también se puede escribir de la forma (ver demostración en anexo A.3):

$$\left|\beta_{00}\right\rangle = \frac{1}{\sqrt{2}}(\left|++\right\rangle + \left|--\right\rangle) \qquad \textbf{(2.6)}$$

Las medidas de Alice de los primeros qubits del par EPR $\left|\beta_{00}\right\rangle$ provocan un colapso instantáneo del estado en uno de los qubits $\left|0\right\rangle$, $\left|1\right\rangle$, $\left|+\right\rangle$ o $\left|-\right\rangle$.

Si Alice usa la base Z para su primera medida en $\left|\beta_{00}\right\rangle$, obtiene los estados cuánticos $\left|0\right\rangle$ o $\left|1\right\rangle$, y los bits clásicos $a'_k = 0$ y $a'_k = 1$, respectivamente.

Si Alice usa la base X para su primera medida en $\left|\beta_{00}\right\rangle$, obtiene los estados cuánticos $\left|+\right\rangle$ y $\left|-\right\rangle$, y los bits clásicos $a'_k = 0$ y $a'_k = 1$, respectivamente.

Bob realiza la medida sobre la pareja del fotón entrelazado que comparte con Alice y que ya ha colapsado en una de las dos posibilidades. Según su elección de bases tendremos las siguientes posibilidades:

1. Mide con Z el qubit $\left|0\right\rangle$ y obtiene el bit 0 con un 100% de probabilidad.
2. Mide con Z el qubit $\left|1\right\rangle$ y obtiene el bit 1 con un 100% de probabilidad.
3. Mide con Z el qubit $\left|+\right\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
4. Mide con Z el qubit $\left|-\right\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
5. Mide con X el qubit $\left|0\right\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
6. Mide con X el qubit $\left|1\right\rangle$ y obtiene el bit 0/1 con un 50% de probabilidad.
7. Mide con X el qubit $\left|+\right\rangle$ y obtiene el bit 0 con un 100% de probabilidad.
8. Mide con X el qubit $\left|-\right\rangle$ y obtiene el bit 1 con un 100% de probabilidad.

A la hora de realizar un intercambio de información por medio de un canal público, el protocolo EPR es muy similar al BB84. Ambos comparan las bases que usaron y descartan todos los fotones con una base “incorrecta” conociendo, previamente, las posiciones donde $a_k = b_k$. La diferencia, sin embargo, es que Alice y Bob deben compartir de antemano una gran cantidad de estados entrelazados. Esto les permite realizar acciones independientes, no importa quien sea el primero en realizar la medida o incluso si ambos la realizan al mismo tiempo. De hecho, el resultado de sus mediciones es independiente del orden de sus secuencias. Finalmente, la eficiencia estimada en este protocolo es por promedio del 50%, de igual modo que en el protocolo BB84.

Para ver un ejemplo de sesión de distribución de clave cuántica para el protocolo EPR consultar el anexo A.2.

CAPÍTULO 3. SIMULACIÓN Y ANÁLISIS DE LA SESIÓN QKD

En este capitulo, se estudia el comportamiento de los diferentes protocolos de distribución de clave cuántica, alterados por los efectos introducidos por el canal cuántico, errores de medida añadidos por los detectores de fotones (*single photon detectors*) y/o también por las acciones de manipulación durante la comunicación.

Para realizar la simulación, el análisis y la representación de gráficos de los tres protocolos se ha utilizado el software *MATLAB*, ampliamente conocido y utilizado en universidades para el aprendizaje en cursos básicos y avanzados de matemáticas, ciencias y, especialmente, ingeniería. Esta herramienta es especialmente útil para el estudio propuesto dado que el tipo de dato básico que gestiona es una matriz o vector (*array*).

Para realizar el desarrollo de los protocolos, se trabajará con ficheros de procesamiento de comandos, también conocidos como ficheros *script*. Un fichero *script* es una secuencia de comandos *MATLAB* que se ejecutan en el orden en que éstos han sido escritos (igual que si se ejecutarán uno a uno en la Ventana de Comandos). La utilización de este tipo de ficheros es conveniente, ya que pueden ser editados (es decir, se pueden corregir o modificar), y se pueden ejecutar tantas veces como se quiera (ver [4]).

El *script* programado (QKD.m) nos permiten simular, en función de un conjunto de parámetros de entrada, una sesión de distribución de clave cuántica (*QKD*) para los protocolos BB84, B92 y EPR.

3.1. Variables de entrada del *script* QKD

Cuando se ejecuta el fichero *script*, las variables utilizadas en los cálculos dentro del fichero deben tener valores asignados previamente. La asignación de valores a estas variables se realiza en la Ventana de Comandos de *MATLAB*.

Una vez ejecutado el fichero, la Ventana de Comandos tendrá el aspecto mostrado en la figura 3.1.

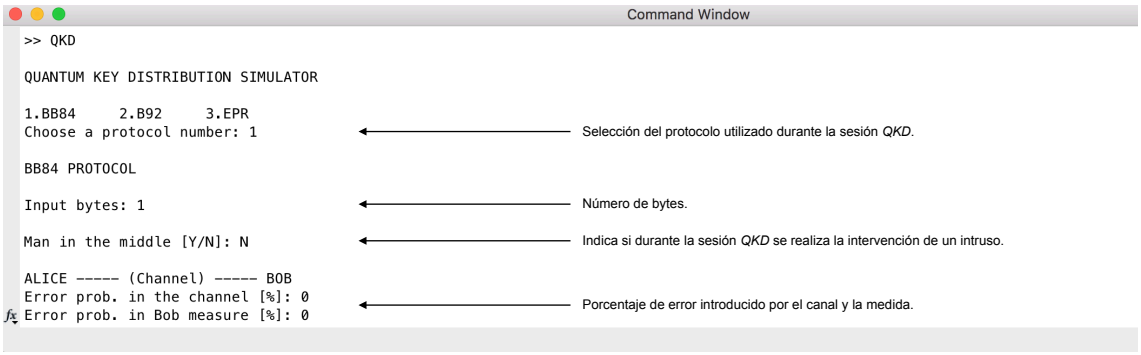

Fig. 3.1 Variables de entrada en la Ventana de Comandos de *MATLAB*

El primer parámetro que tenemos que introducir es el número asociado al protocolo que queremos utilizar durante la simulación de la sesión *QKD*. Los comandos aceptados por el *script* son “1” (protocolo BB84), “2” (protocolo B92) y “3” (protocolo EPR).

A continuación, introducimos el número de bytes utilizados para generar la secuencia de bits clásicos aleatoria generada por el emisor (protocolo asimétrico) o bien, por el emisor y el receptor (protocolo simétrico).

Por último, es necesario definir si durante la comunicación se realizará una intervención de un intruso (*man in the middle* o Eva) actuando sobre la secuencia enviada. Los comandos aceptados por el *script* son “Y” (Yes) o “N” (No). En función del carácter introducido, el escenario de la sesión *QKD* generado será distinto.

Para el caso donde no participa un intruso (“N”), la comunicación se realiza entre emisor y receptor (Alice y Bob), por un canal clásico (o público) y un canal cuántico donde podrá actuar una probabilidad de error ε_{ch1} que simula el comportamiento de un medio de transmisión como puede ser, por ejemplo, la fibra óptica o el espacio libre. Además, podemos introducir una probabilidad de error ε_{m1} para simular el grado de error durante la medida de los fotones mediante el detector (*single photon detectors*) de Bob. La figura 3.2 muestra el escenario descrito:

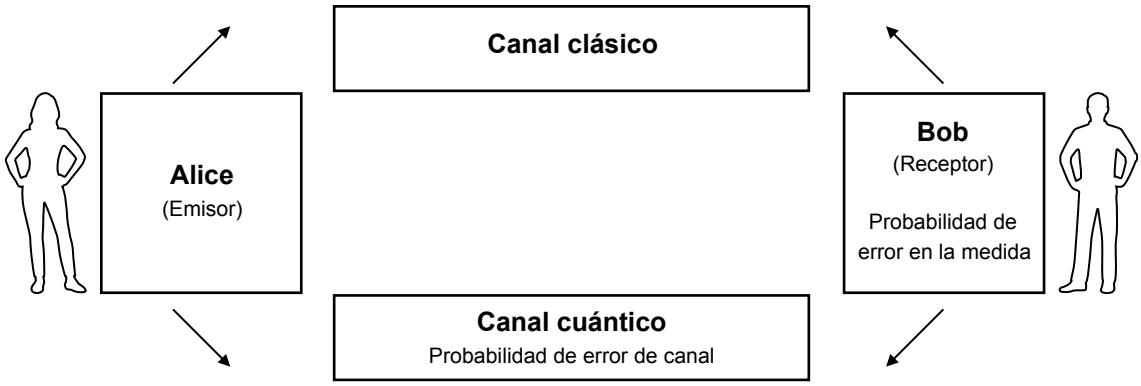

Fig. 3.2 Esquema de comunicación entre Alice y Bob (sin intruso)

Para el caso donde sí participa un intruso ("Y"), la comunicación entre Alice y Bob es interceptada por Eva. Es posible que el medio de transmisión del canal cuántico presente características distintas durante la comunicación entre Alice y Eva y, los introducidos entre Eva y Bob. Por tanto, actuarán dos probabilidades de error que llamaremos ε_{ch1} y ε_{ch2}, respectivamente, para distinguir ambas comunicaciones. Además, tanto Eva como Bob tendrán una probabilidad de error de medida ε_{m1} y ε_{m2}, respectivamente, que aplicará a la hora de recibir los fotones. La figura 3.3 muestra el escenario descrito:

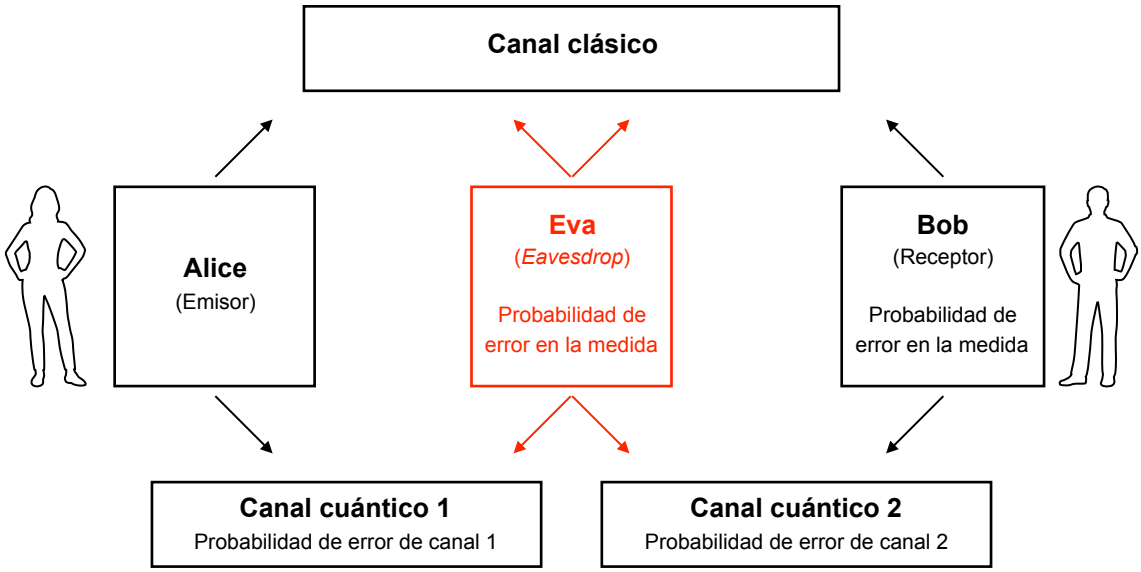

Fig. 3.3 Esquema de comunicación entre Alice y Bob interceptada por Eva

3.2. Variables de salida del *script* QKD

MATLAB genera automáticamente la salida de la ejecución de nuestro código simulado y muestra directamente el resultado en la Ventana de Comandos.

Una vez introducidas las variables de entrada del fichero, la Ventana de Comandos tendrá el aspecto mostrado en la figura 3.4.

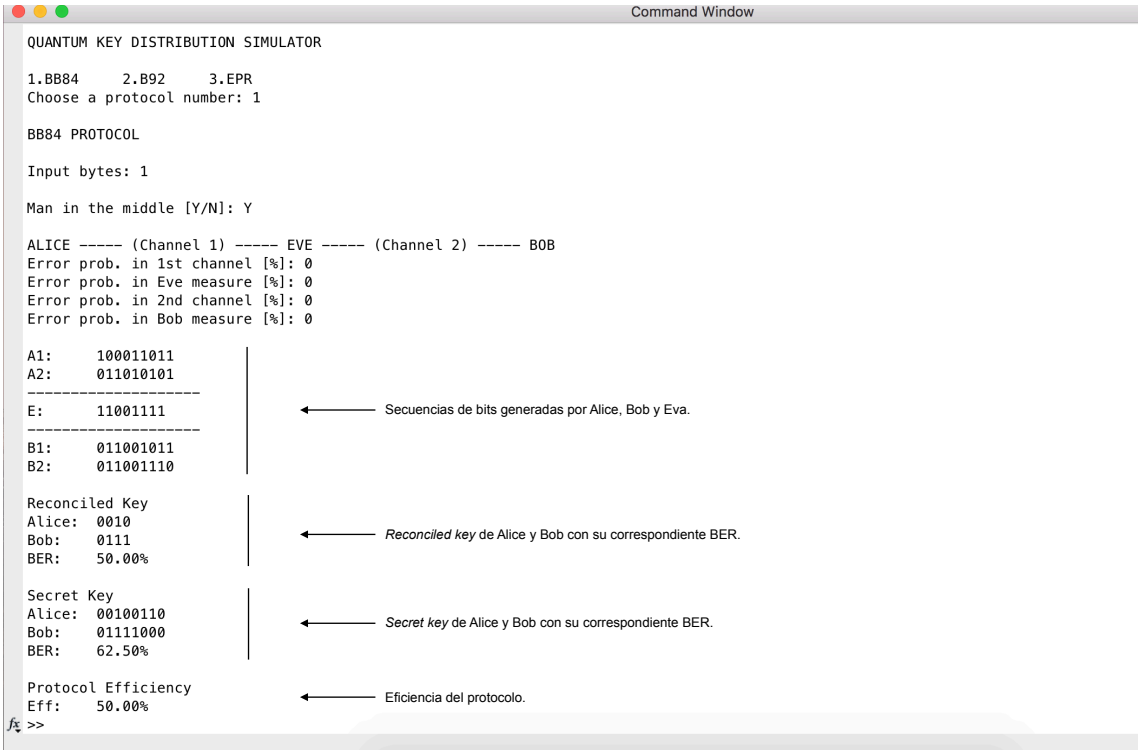

Fig. 3.4 Variables de salida en la Ventana de Comandos de *MATLAB*

La Venta de Comando, una vez generada la salida de la ejecución, muestra las secuencias de bits (*arrays*): A (Alice), B (Bob) y, si lo requiere E (Eva). En estas secuencias se indica en formato binario la base de medida o la polarización, siguiendo los convenios definidos anteriormente en cada uno de los protocolos.

Para agregar robustez a la comunicación, cada una de las secuencias de longitud N iniciales disponen de un bit de paridad en la posiciones $N+1$. Este mecanismo de detección de errores indica si el número de bits con valor 1 en un conjunto de bits es par o impar. En el caso donde el número de bits con valor 1 sea par, el bit de paridad será 1. Por otro lado, si el número de bits con valor 1 es impar, el bit de paridad será 0. Nótese que este método detecta errores, pero no los corrige.

A continuación, tenemos el valor de la *reconciled key* y la *secret key* de Alice y Bob con su respectiva tasa de error binario (BER) que obtenemos a partir de la ecuación descrita en (2.2). En el caso que no sea posible generar ninguna clave el *script* vuelve a ejecutarse automáticamente. Por último, se muestra el valor de la eficiencia del protocolo definida en la ecuación (2.3).

3.3. Sesión QKD: protocolo BB84

El *script* desarrollado nos permite simular y analizar el comportamiento del protocolo BB84 durante una sesión de distribución de clave cuántica (*QKD*).

Se estudiará la tasa de error binario (BER) tanto de la *reconciled key* como de la *secret key*, el impacto de los errores introducidos por el canal cuántico y los instrumentos de medida, las acciones de manipulación de Eva y, por último, la eficiencia del protocolo. Todas las ejecuciones se realizarán con un total de 1000 repeticiones (muestras) para obtener un valor representativo en relación al coste computacional de cada operación.

3.3.1. Estudio de la tasa de error binario en función del número de bits

El objetivo de este análisis es estudiar la diferencia entre la tasa de error binario (BER) de la *reconciled key* y la *secret key*.

Las ejecuciones se realizarán con las siguientes variables de entrada:

1. Protocolo: BB84.
2. Bytes: 20, 80, 320, 640 y 1280 bytes.
3. Comunicación directa entre Alice y Bob (Fig. 3.2).
4. Probabilidad de error de canal ε_{ch1} (Alice y Bob) y medida ε_{m1} (Bob): 20% y 0% (respectivamente).

El gráfico 3.5 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* en función del número de bytes:

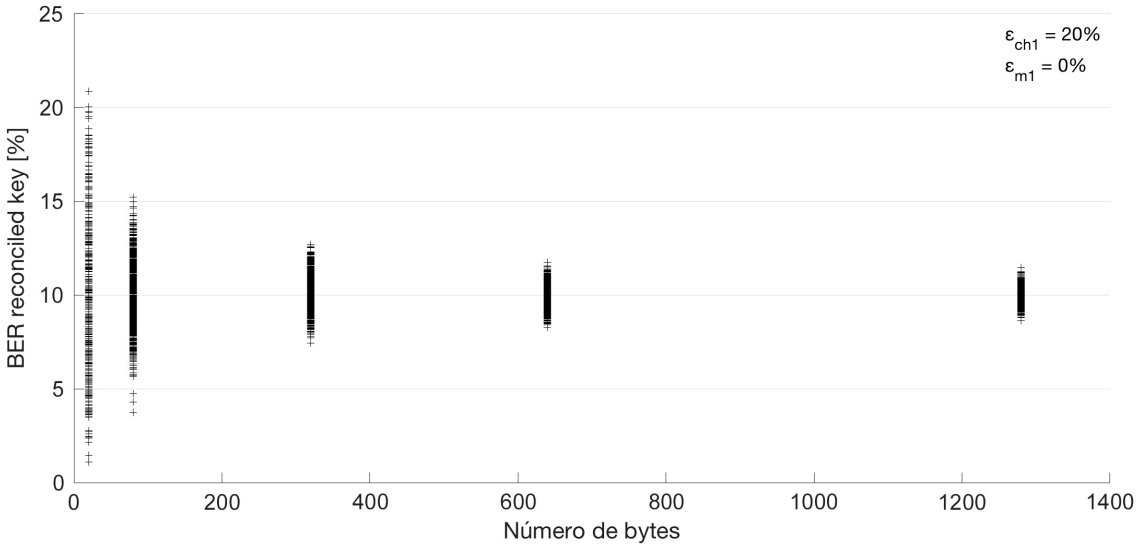

Fig. 3.5 BER de la *reconciled key* en función del número de bytes

La BER de la *reconciled key* presenta una menor dispersión de valores para un mayor número de bytes. Con secuencias de bits mas largas los resultados son menos dispares y, por tanto, podemos observar la tendencia que sigue el error. La distribución de los valores de la BER obtenida a partir de 1000 ejecuciones diferentes, nos permite analizar de forma representativa, la media aritmética $\bar{X}$ y la desviación estándar σ respectivamente:

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n}, \text{ donde } n = 1000 \text{ muestras} \quad \textbf{(3.1)}$$

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} (X_i - \bar{X})^2}{n}} \quad \textbf{(3.2)}$$

El resultado que obtenemos para 1280 bytes tras realizar la ejecución automática durante 1000 repeticiones es de $\bar{X} = 0.10002$ y $\sigma = 0.0043$, es decir, $[10.0 \pm 0.5]\%$.

Con una probabilidad de error de canal (ε_{ch1}) del 20%, podemos esperar que 1 de cada 5 bits serán generados aleatoriamente con un 50% de probabilidades de ser correctos o erróneos, con respecto al bit enviado por el emisor. Por tanto, podemos concluir que en un 10% de los casos, el bit resultante será erróneo. El resultado de este análisis teórico coincide con el obtenido tras realizar el estudio estadístico de la ejecución del *script* con *MATLAB*. Esto nos permite concluir que, en función del coste computacional (incremento de tiempo por simulación), utilizar 1280 bytes es suficientemente preciso (error a partir del tercer decimal) como para obtener resultados representativos.

Ampliando la privacidad de la clave, el valor medio de la BER cabe esperar que será más elevado. El error acumulado se propaga a la *secret key* siguiendo, en nuestro caso, el operador lógico XOR (Tabla 2.3).

El gráfico 3.6 nos muestra la variación de la tasa de error binario (BER) de la *secret key* en función del número de bytes:

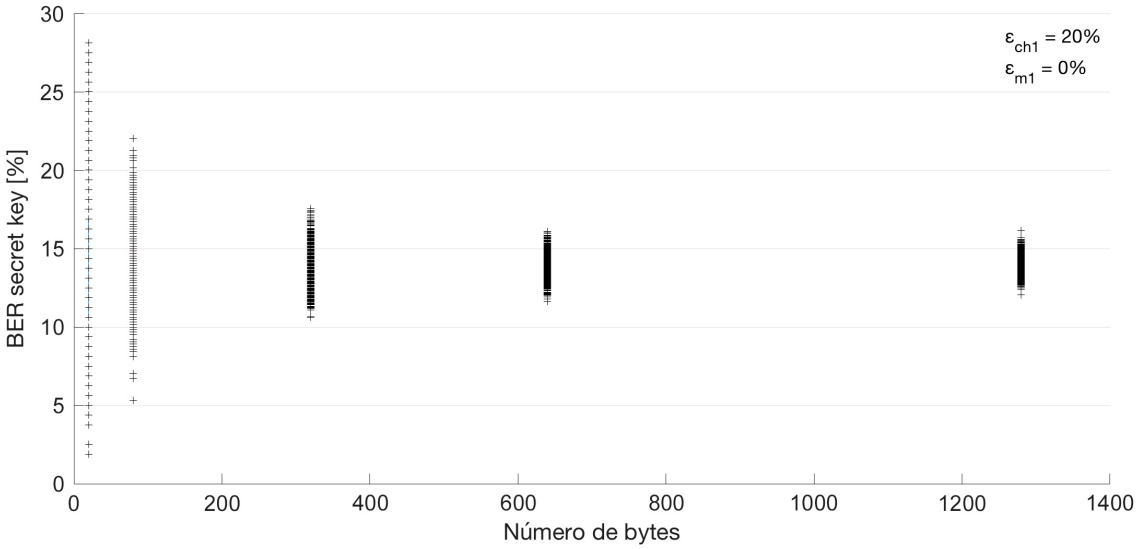

Fig. 3.6 BER de la secret *key* en función del número de bytes

Como en el caso anterior, se realiza un estudio estadístico de la media aritmética $\bar{X}$ (3.3) y la desviación estándar (3.4) de la BER para la secuencia compuesta por 1280 bytes:

El resultado que obtenemos para 1280 bytes tras realizar la ejecución automática durante 1000 repeticiones es de $\bar{X} = 0.14$ y $\sigma = 0.0032$, es decir, $[14.0 \pm 0.4]\%$.

Así pues, podemos comprobar que la media aritmética de la BER es mayor cuando aplicamos una ampliación de la privacidad en la clave generada. Aumentar la complejidad y, por tanto, la seguridad de la clave conlleva también aumentar los errores producidos durante la comunicación.

3.3.2. Estudio del error introducido por el canal cuántico

El objetivo de este análisis es estudiar la evolución del error introducido por el canal cuántico (ε_{ch1}) en función de la tasa de error binario (BER) de la *reconciled key* y la *secret key* durante una comunicación directa entre Alice y Bob y durante una comunicación con Eva interceptando.

La primera ejecución se realizará con las siguientes variables de entrada:

1. Protocolo: BB84.
2. Bytes: 1280 bytes.
3. Comunicación directa entre Alice y Bob (Fig. 3.2).

4. Probabilidad de error de canal ε_{ch1} (Alice y Bob) y medida ε_{m1} (Bob): [0, 100]% y 0% (respectivamente).

El gráfico 3.7 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error en el canal cuántico (ε_{ch1}) sin error en la medida ($\varepsilon_{m1} = 0\%$) durante una comunicación directa entre Alice y Bob (sin intruso):

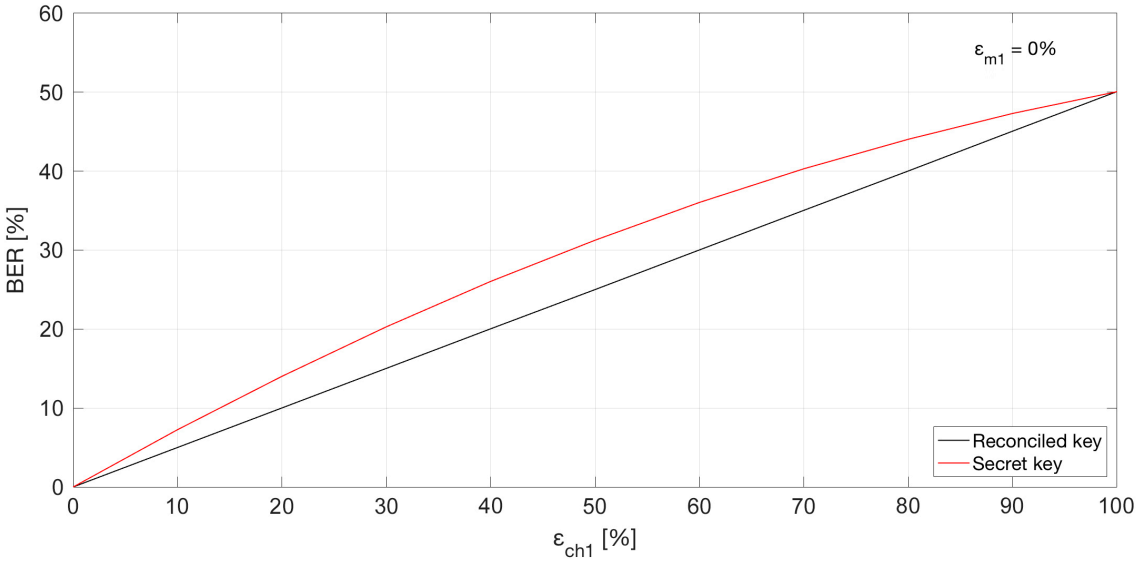

Fig. 3.7 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} (para $\varepsilon_{m1} = 0\%$)

El comportamiento de la BER para la *reconciled key* sigue una evolución lineal. Dado que el tipo de dato básico que se utiliza es un vector (array) de “0” y “1”, cada bit trasmitido por el canal con una probabilidad de error ε_{ch1} es recibido erróneamente con una probabilidad $\varepsilon_{ch1}/2$, por tanto, con una $BER \simeq \varepsilon_{ch1}/2$. Para la BER de la *secret key* tenemos una evolución que presenta una máxima diferencia para $\varepsilon_{ch1} = 50\%$, con respecto a la *reconciled key*. A partir de este punto, decrece gradualmente hasta alcanzar el mismo valor de BER para ambas claves en $\varepsilon_{ch1} = 100\%$.

Comparando ambas claves, podemos distinguir un factor que amplifica la BER de la *secret key* con respecto la de la *reconciled key* hasta $\varepsilon_{ch1} = 50\%$. Este factor corresponde al error propagado por el operador lógico XOR. Para la máxima diferencia se obtiene que la amplificación de privacidad es aproximadamente de 1.51 ($BER_{secretkey} \simeq 1.51 \cdot BER_{reconciledkey}$).

A continuación, se realizará una segunda ejecución en la que introducimos a Eva, con las siguientes variables de entrada:

1. Protocolo: BB84.
2. Bytes: 1280 bytes.
3. Comunicación entre Alice y Bob interceptada por Eva (Fig. 3.3).
4. Probabilidad de error de canal ε_{ch1} (Alice y Eva) y ε_{ch2} (Eva y Bob): [0, 100]%.
5. Probabilidad de error de medida ε_{m1} (Eva) y ε_{m2} (Bob): 0%.

El gráfico 3.8 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{ch2} (para $\varepsilon_{ch1} = \varepsilon_{ch2}$) durante una comunicación entre Alice y Bob interceptada por Eva:

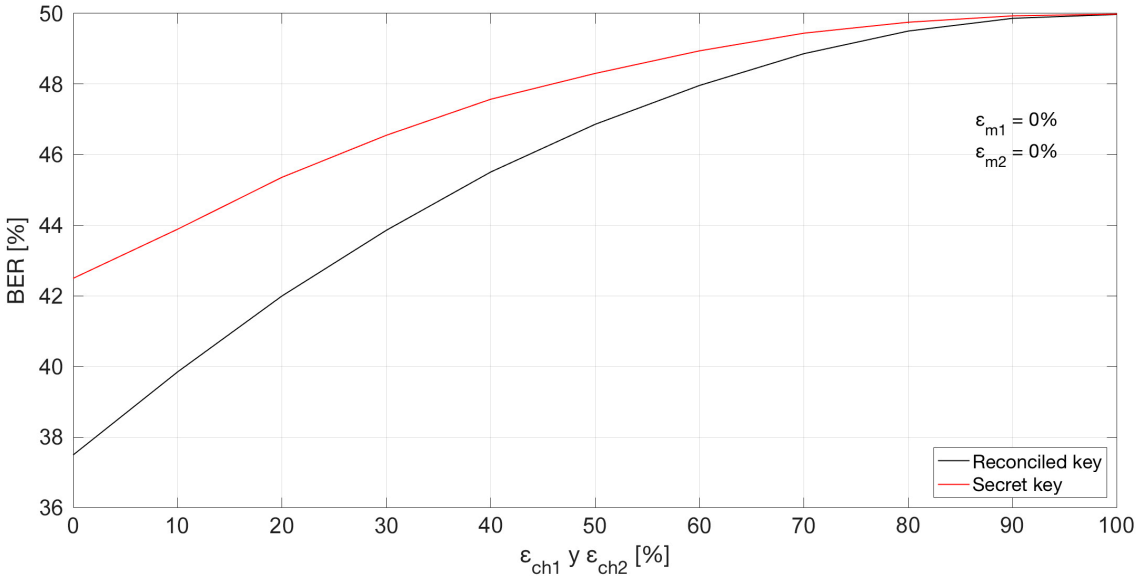

Fig. 3.8 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{ch2} (para $\varepsilon_{ch1} = \varepsilon_{ch2}$)

Inicialmente, para un error $\varepsilon_{ch1} = \varepsilon_{ch2} = 0\%$ y $\varepsilon_{m1} = \varepsilon_{m2} = 0\%$, observamos una $BER \simeq 37.5\%$ para la *reconciled key* y una $BER \simeq 42.5\%$ para la *secret key*. Por tanto, podemos concluir que la medición que realiza Eva, durante la comunicación entre Alice y Bob por el canal cuántico, sí es detectada sin la participación de ningún error externo. Este resultado pone de manifiesto el principio cuántico, de que el observador, al realizar una medida, modifica lo observado.

La evolución de la BER de ambas claves crece en función del error introducido, partiendo de una ordenada en el origen distinta (debido al mecanismo de amplificación de privacidad), hasta converger en $\varepsilon_{ch1} = \varepsilon_{ch2} = 100\%$.

3.3.3. Estudio del error introducido por la medida

El objetivo de este análisis es estudiar la evolución del error introducido por el canal de forma conjunta al error de medida en función de la tasa de error binario (BER) de la *reconciled key* y la *secret key,* durante una comunicación directa entre Alice y Bob y durante una comunicación con Eva interceptando.

La primera ejecución se realizará con las siguientes variables de entrada:

1. Protocolo: BB84.
2. Bytes: 1280 bytes.
3. Comunicación directa entre Alice y Bob (Fig. 3.2).
4. Probabilidad de error de canal ε_{ch1} (Alice y Bob) y medida ε_{m1} (Bob): [0, 100]% y [0, 100]% (respectivamente).

El gráfico 3.9 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{m1} y ε_{m1} (para $\varepsilon_{ch1} = \varepsilon_{m1}$) durante una comunicación directa entre Alice y Bob (sin intruso):

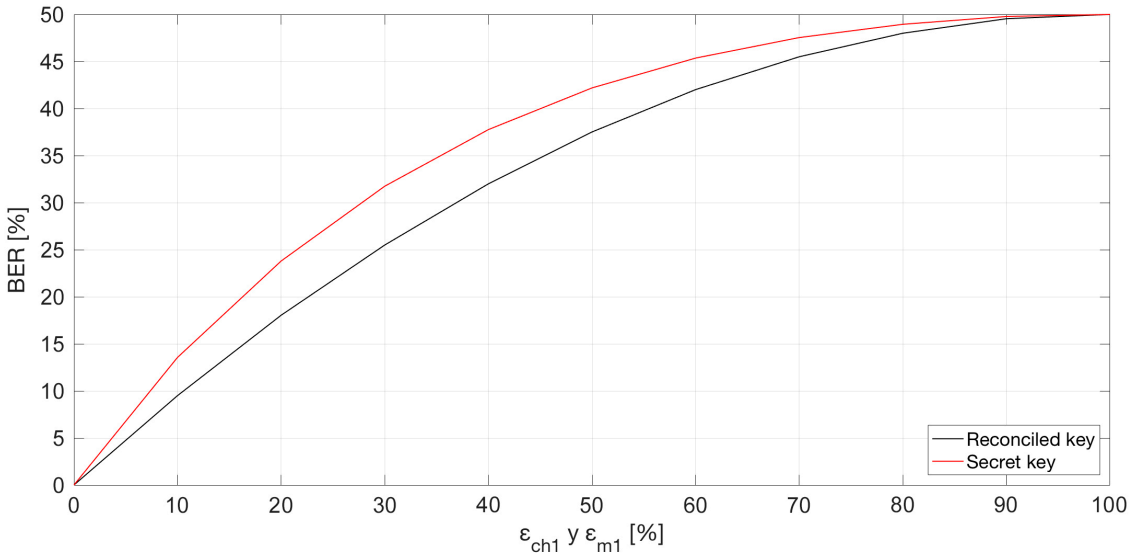

Fig. 3.9 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{m1} (para $\varepsilon_{ch1} = \varepsilon_{m1}$)

En comparación con el primer estudio sobre el error introducido por el canal (Fig. 3.7), ambas claves presentan una BER mas elevada para un error dado. De esta forma, podemos observar como la medición de Bob aumenta el error del sistema simulando el error introducido por los instrumentos de medida (*single photon detectors*).

El comportamiento de la BER en función del error ε_{ch1} o ε_{m1} para un valor constante de error $\varepsilon_{m1} = 0\%$ o $\varepsilon_{ch1} = 0\%$, respectivamente, es el mismo.

A continuación, se realizará una segunda ejecución en la que introducimos a Eva, con las siguientes variables de entrada:

1. Protocolo: BB84.
2. Bytes: 1280 bytes.
3. Comunicación entre Alice y Bob interceptada por Eva (Fig. 3.3).
4. Probabilidad de error de canal ε_{ch1} (Alice y Eva) y ε_{ch2} (Eva y Bob): [0, 100]%.
5. Probabilidad de error de medida ε_{m1} y ε_{m2}: [0, 100]%.

El gráfico 3.10 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{ch1}, ε_{ch2}, ε_{m1} y ε_{m2} (para $\varepsilon_{ch1} = \varepsilon_{ch2} = \varepsilon_{m1} = \varepsilon_{m2}$) durante una comunicación entre Alice y Bob interceptada por Eva:

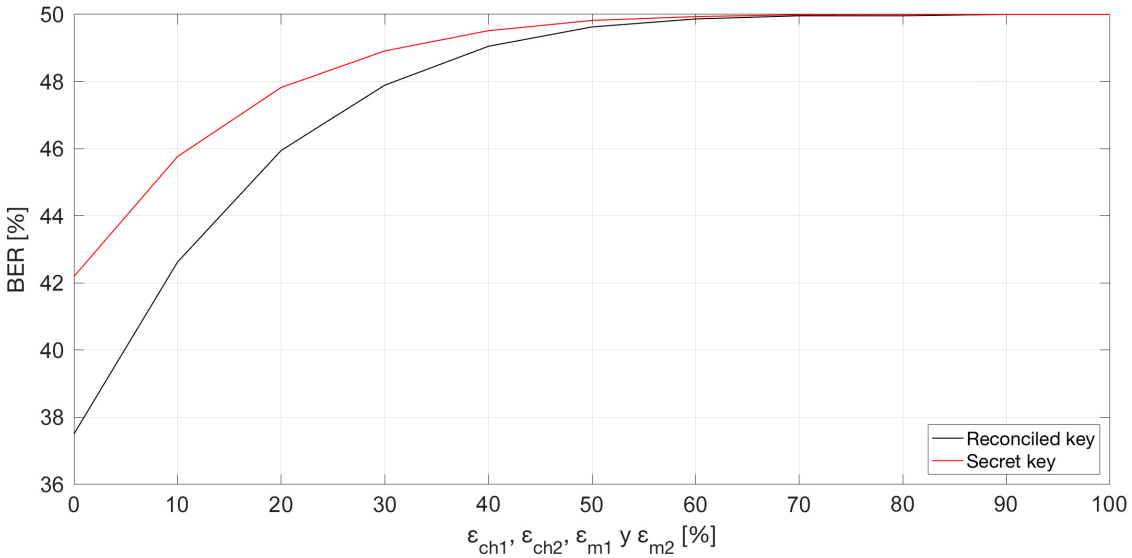

Fig. 3.10 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1}, ε_{ch2}, ε_{m1} y ε_{m2} (para $\varepsilon_{ch1} = \varepsilon_{ch2} = \varepsilon_{m1} = \varepsilon_{m2}$)

Inicialmente, para un error $\varepsilon_{ch1} = \varepsilon_{ch2} = 0\%$ y $\varepsilon_{m1} = \varepsilon_{m2} = 0\%$, observamos como es lógico, la misma BER que para el segundo estudio sobre el error introducido por el canal (Fig. 3.8).
La ordenada en el origen no se ve modificada pero si la pendiente, provocando que ambas claves converjan a una $BER = 50\%$. A partir de ese punto, la amplificación de privacidad sobre la *secret key* es nula con respecto la *reconciled* key y no existe diferencia entre la BER de ambas claves, debido a que han alcanzado el umbral máximo de error ($BER = 50\%$).

3.3.4. Estudio de la eficiencia

El objetivo de este análisis es estudiar la evolución del error introducido por el canal de forma conjunta al error de medida en función de la eficiencia del sistema, durante una comunicación directa entre Alice y Bob y durante una comunicación con Eva interceptando.

El gráfico 3.11 nos muestra la variación de la eficiencia (Eff) para las variaciones estudiadas de probabilidad de error de canal y medida, durante una comunicación directa entre Alice y Bob y con la intervención de Eva:

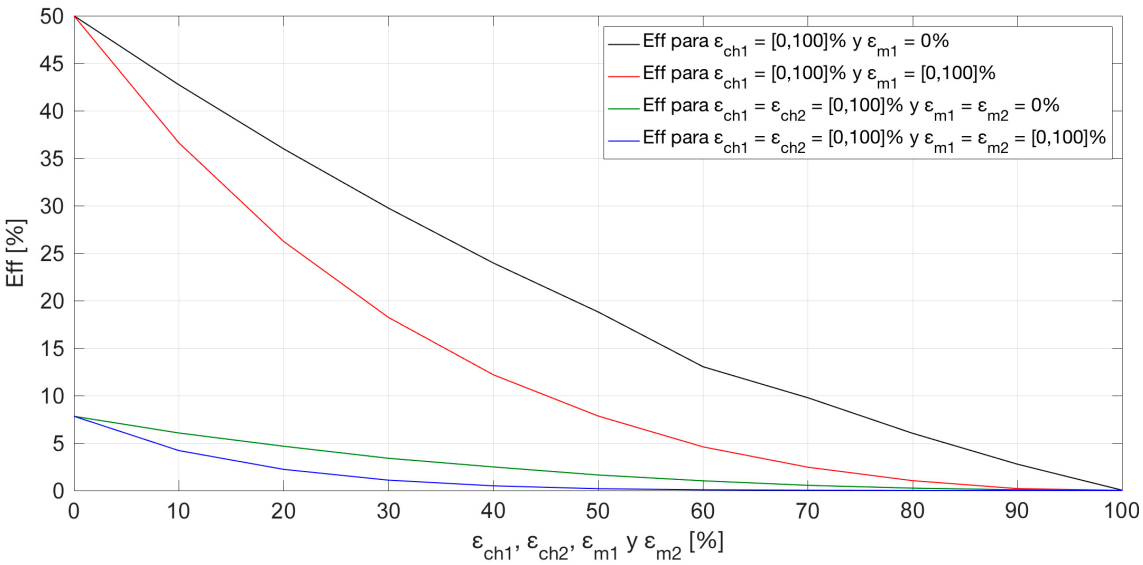

Fig. 3.11 Eficiencia (Eff) del protocolo BB84

El protocolo BB84, durante una comunicación directa entre Alice y Bob para un error de canal y de medida nulo (condiciones ideales), presenta una $Eff = 50\%$. Las curvas negra y roja, muestran la evolución de la eficiencia (Eff) para la variación del error de canal entre Alice y Bob (ε_{ch1}) y la medida de

Bob (ε_{m1}). Por otra parte, las curvas verde y azul son el resultado de la intervención de Eva. La ordenada en el origen (error de canal y de medida nulo) indica una $Eff = 7.82\%$. Por tanto, la participación de un intruso durante una sesión de distribución clave cuántica (*QKD*) para el protocolo BB84 disminuye la eficiencia (Eff) en, aproximadamente, un 42.18% con respecto una comunicación directa entre el emisor y receptor original en el caso ideal.

3.4. Sesión QKD: protocolo B92

El *script* desarrollado nos permite simular y analizar el comportamiento del protocolo B92 durante una sesión de distribución clave cuántica (*QKD*).

Se estudiará el impacto de los errores introducidos por el canal cuántico y los instrumentos de medida, las acciones de manipulación de Eva y, por último, la eficiencia del protocolo.

3.4.1. Estudio del error introducido por el canal cuántico

La primera ejecución se realiza con las siguientes variables de entrada:

1. Protocolo: B92.
2. Bytes: 1280 bytes.
3. Comunicación directa entre Alice y Bob (Fig. 3.2).
4. Probabilidad de error de canal ε_{ch1} (Alice y Bob) y medida ε_{m1} (Bob): [0, 100]% y 0% (respectivamente).

El gráfico 3.12 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{ch1} (para $\varepsilon_{m1} = 0\%$) durante una comunicación directa entre Alice y Bob (sin intruso):

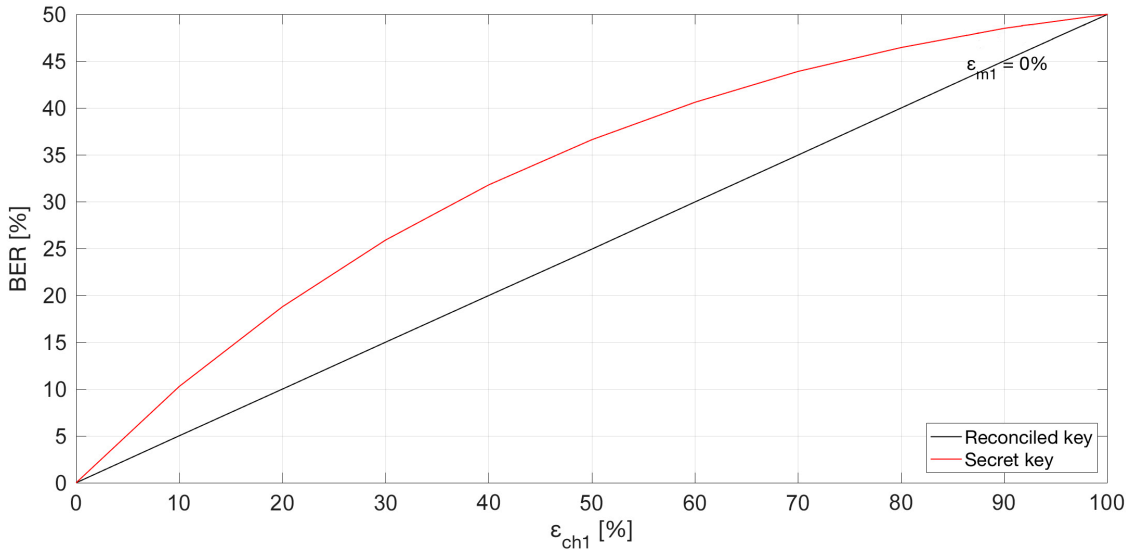

Fig. 3.12 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} (para $\varepsilon_{m1} = 0\%$)

Con respecto al mismo escenario para el protocolo BB84 (Fig. 3.7), la BER de la *reconciled key* sigue el mismo comportamiento lineal. El emisor recibe una secuencia de bits con una probabilidad de error $\varepsilon_{ch1}/2$, por tanto, con una $BER \simeq \varepsilon_{ch1}/2$. Comparando ambas claves, podemos distinguir un factor que amplifica la BER de la *secret key* con respecto la de la *reconciled key* siendo la máxima diferencia nuevamente $\varepsilon_{ch1} = 50\,\%$. Así mismo, se obtiene que la amplificación de privacidad es mayor que la del protocolo BB84, con un valor aproximado de 1.72 ($BER_{secretkey} \simeq 1.72 \cdot BER_{reconciledkey}$). Esto se debe a que el número de bits utilizados para generar la *reconciled key* (bits útiles) es un 50% menor que para el BB84. Cuanto más corta sea la *reconciled key* con respecto al tamaño de la secuencia aleatoria inicial, más se propagará el error por medio del operador lógico XOR al generar la *secret key*.

A continuación, se realizará una segunda ejecución en la que introducimos a Eva, con las siguientes variables de entrada:

1. Protocolo: B92.
2. Bytes: 1280 bytes.
3. Comunicación entre Alice y Bob interceptada por Eva (Fig. 3.3).
4. Probabilidad de error de canal ε_{ch1} (Alice y Eva) y ε_{ch2} (Eva y Bob): [0, 100]%.
5. Probabilidad de error de medida ε_{m1} (Eva) y ε_{m2} (Bob): 0%.

El gráfico 3.13 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{ch2} (para $\varepsilon_{ch1} = \varepsilon_{ch2}$) durante una comunicación entre Alice y Bob interceptada por Eva:

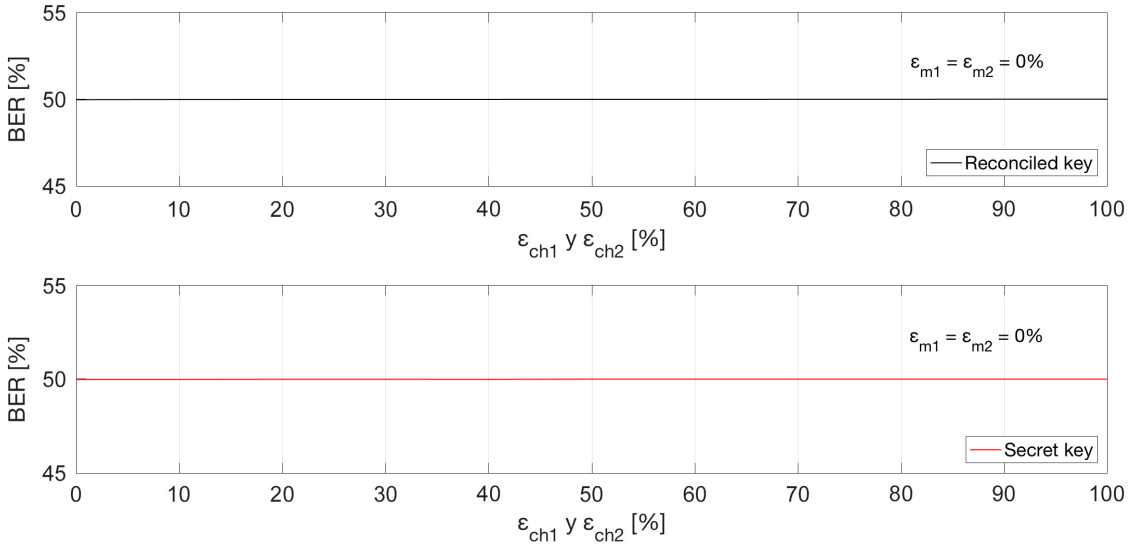

Fig. 3.13 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{ch2} (para $\varepsilon_{ch1} = \varepsilon_{ch2}$)

Para ambas claves, tenemos un $BER = 50\%$ (umbral máximo de error para una secuencia binaria) de forma constante para todo el rango de valores de error de canal ($\varepsilon_{ch1} = \varepsilon_{ch2} = [0,100]\%$), esto es debido a que la medida de Eva produce una $BER = 50\%$.

En el protocolo BB84, la participación de Eva añade una $BER \simeq 37.5\%$, como se trata de una $BER < 50\%$, el error externo (canal y medida) sí añade un aumento de BER en ambas claves hasta que convergen para $BER = 50\%$. El protocolo B92, en cambio, al tratarse de un versión simplificada que solo utiliza los fotones polarizados $|0\rangle$ y $|+\rangle$, cualquier tipo de modificación en la secuencia recibida por Bob (incluyendo la medida de Eva) respecto a la enviada por Alice, provoca un aumento mucho mayor de la BER. Esto implica que la participación de error externo no aumente la BER en ninguna de las dos claves para un valor de error dado.

Por tanto, podemos concluir que la medición que realiza Eva, durante la comunicación entre Alice y Bob por el canal cuántico, se detecta sin la

participación de ningún error externo con una diferencia del 12.5% más $(50 - 37.5\%)$ en relación al protocolo BB84.

Por tanto, se puede concluir que añadiendo el error introducido por la medida de Eva y Bob, se obtendrá el mismo resultado.

2.4.2. Estudio del error introducido por la medida

La ejecución se realiza con las siguientes variables de entrada:

1. Protocolo: B92.
2. Bytes: 1280 bytes.
3. Comunicación directa entre Alice y Bob (Fig. 3.2).
4. Probabilidad de error de canal ε_{ch1} (Alice y Eva) y medida ε_{m1} (Eva): [0, 100]% y [0, 100]% (respectivamente).

El gráfico 3.14 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{m1} y ε_{m1} (para $\varepsilon_{ch1} = \varepsilon_{m1}$) durante una comunicación directa entre Alice y Bob (sin intruso):

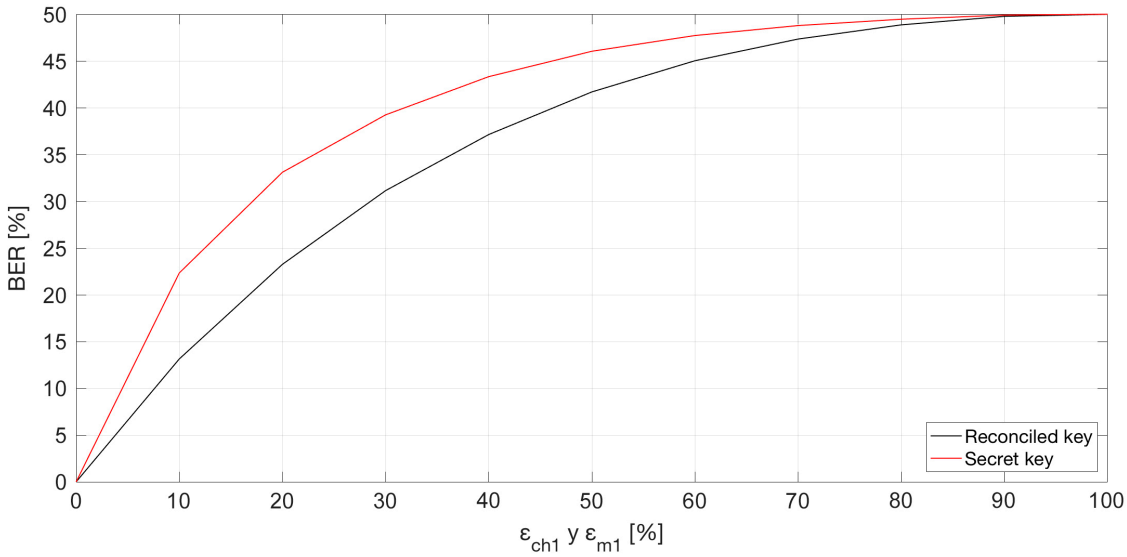

Fig. 3.14 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{m1} (para $\varepsilon_{ch1} = \varepsilon_{m1}$)

Podemos observar como la medición de Bob aumenta el error del sistema simulando el error introducido por los instrumentos de medida (*single photon detectors*). Además, si realizamos una comparación con respecto al protocolo BB84, ambas claves presentan una BER más elevada para todo el rango de

valores de error de canal y de medida. La suma de errores afecta de forma más significativa a la robustez del protocolo, debido a la simplificación del mismo.

2.4.3. Estudio de la eficiencia

El gráfico 3.15 nos muestra la variación de la eficiencia (Eff) para las variaciones estudiadas de probabilidad de error de canal y medida, durante una comunicación directa entre Alice y Bob y con la intervención de Eva:

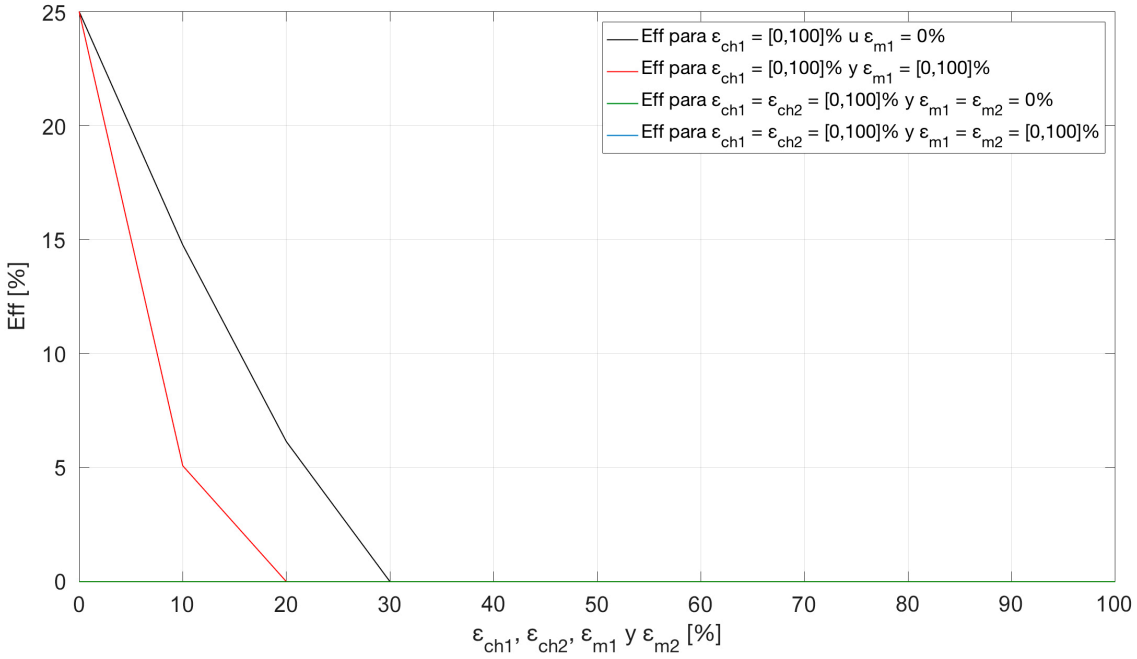

Fig. 3.15 Eficiencia (Eff) del protocolo B92

El protocolo B92, durante una comunicación directa entre Alice y Bob para un error de canal y de medida nulo (condiciones ideales), presenta una $Eff = 25\%$. Las curvas negra y roja, muestran la evolución de la eficiencia (Eff) para la variación del error de canal entre Alice y Bob (ε_{ch1}) y la medida de Bob (ε_{m1}). Por otra parte, las curvas verde y azul (solapadas en el eje de abscisas) son el resultado de la intervención de Eva, indicando una $Eff = 0\%$. Esto es debido a que la BER alcanza el umbral máximo (50%) desde el inicio de la intervención. Por tanto, la participación de un intruso durante una sesión de distribución clave cuántica (*QKD*) para el protocolo B92 disminuye la eficiencia (Eff) en, aproximadamente, un 25% con respecto una comunicación directa entre el emisor y receptor original.

3.5. Sesión QKD: protocolo EPR

El *script* desarrollado nos permite simular y analizar el comportamiento del protocolo EPR durante una sesión de distribución clave cuántica (*QKD*).

Se estudiará el impacto de los errores introducidos por el canal cuántico y los instrumentos de medida, las acciones de manipulación de Eva y, por último, la eficiencia del protocolo.

3.5.1. Estudio del error introducido por el canal cuántico

La primera ejecución se realiza con las siguientes variables de entrada:

1. Protocolo: EPR.
2. Bytes: 1280 bytes.
3. Comunicación directa entre Alice y Bob (Fig. 3.2).
4. Probabilidad de error de canal ε_{ch1} (Alice y Bob) y medida ε_{m1} (Bob): [0, 100]% y 0% (respectivamente).

El gráfico 3.16 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{ch1} (para $\varepsilon_{m1} = 0\%$) durante una comunicación directa entre Alice y Bob (sin intruso):

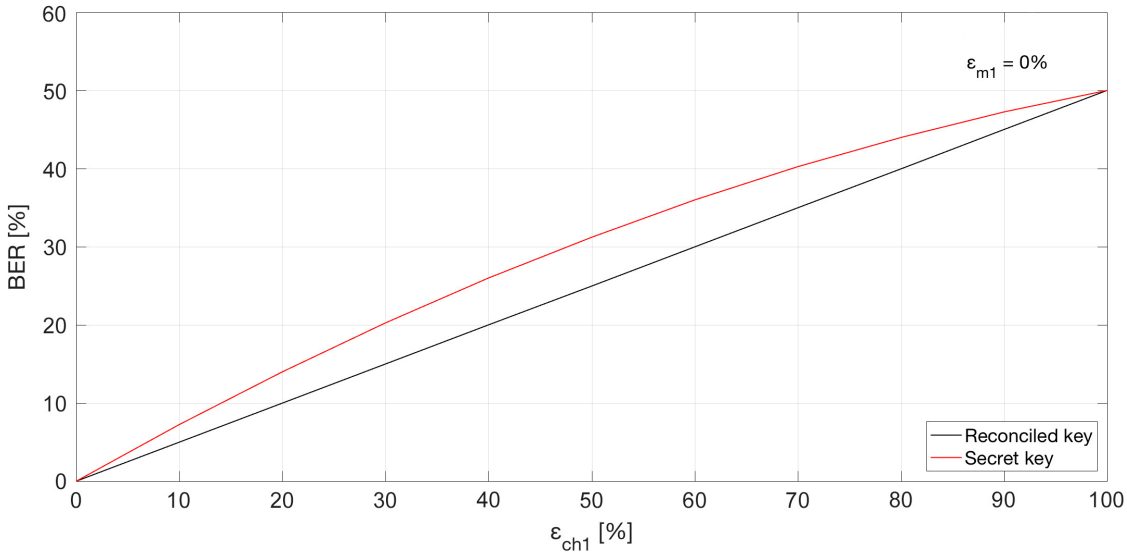

Fig. 3.16 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} (para $\varepsilon_{m1} = 0\%$)

Comparando la BER para la *reconciled key* de los tres protocolos en este escenario, comprobamos como su evolución es lineal y modelable como $BER \simeq \varepsilon_{ch1}/2$, donde ε_{ch1} es la probabilidad de error de canal.

Para la *secret key*, en cambio, tenemos una dependencia directa con el número de bits utilizados para generar la *reconciled key* (bits útiles). Como hemos analizado anteriormente, cuantos menos bits útiles genere el protocolo (nivel de eficiencia) tendrá una mayor BER a causa del error acumulado propagado por el operador lógico XOR.

A continuación, se realizará una segunda ejecución en la que introducimos a Eva, con las siguientes variables de entrada:

1. Protocolo: EPR.
2. Bytes: 1280 bytes.
3. Comunicación entre Alice y Bob interceptada por Eva (Fig. 3.3).
4. Probabilidad de error de canal ε_{ch1} (Alice y Eva) y ε_{ch2} (Eva y Bob): [0, 100]%.
5. Probabilidad de error de medida ε_{m1} (Eva) y ε_{m2} (Bob): 0%.

El gráfico 3.17 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{ch2} (para $\varepsilon_{ch1} = \varepsilon_{ch2}$) durante una comunicación entre Alice y Bob interceptada por Eva:

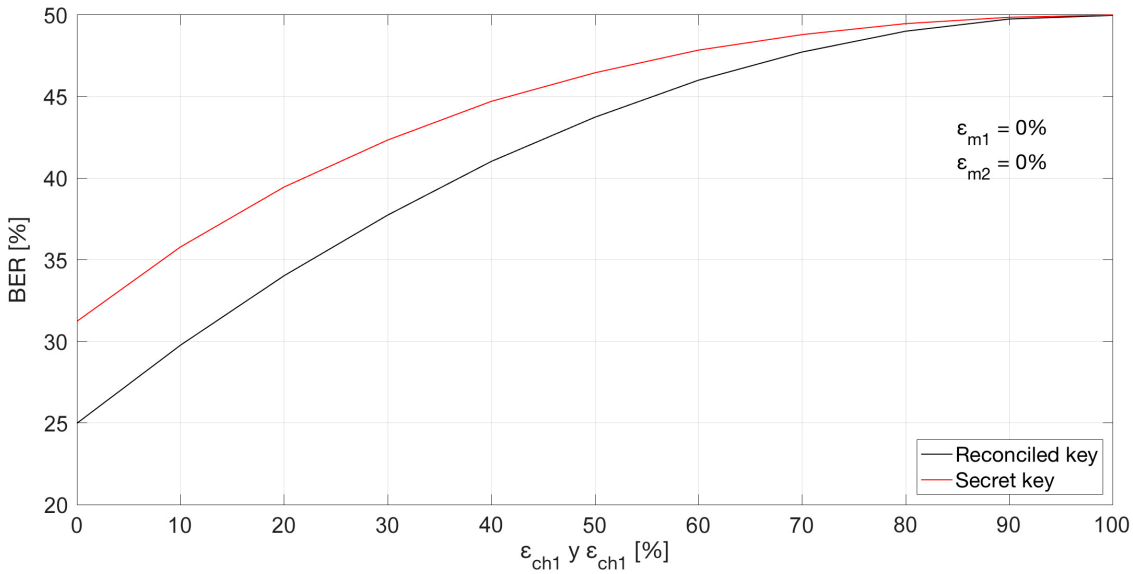

Fig. 3.17 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{ch2} (para $\varepsilon_{ch1} = \varepsilon_{ch2}$)

Como en los anteriores protocolos, podemos concluir que la medición de Eva puede ser detectada si los errores de canal son conocidos. En este caso, tenemos una $BER \simeq 25\%$ para un error $\varepsilon_{ch1} = \varepsilon_{ch2} = 0\%$ y $\varepsilon_{m1} = \varepsilon_{m2} = 0\%$, es decir, sin la participación de ningún error externo.

3.5.2. Estudio del error introducido por la medida

La primera ejecución se realiza con un total de 1000 repeticiones (muestras) y con las siguientes variables de entrada:

1. Protocolo: EPR.
2. Bytes: 1280 bytes.
3. Comunicación directa entre Alice y Bob (Fig. 3.2).
4. Probabilidad de error de canal ε_{ch1} (Alice y Bob) y medida ε_{m1} (Bob): [0, 100]% y [0, 100]% (respectivamente).

El gráfico 3.18 nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{m1} y ε_{m1} (para $\varepsilon_{ch1} = \varepsilon_{m1}$) durante una comunicación directa entre Alice y Bob (sin intruso):

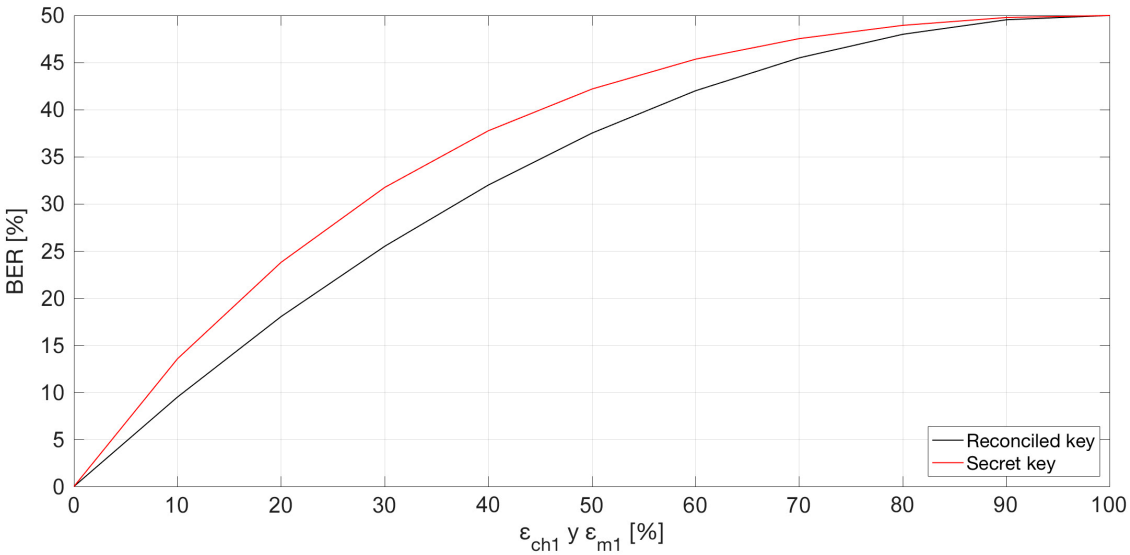

Fig. 3.18 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1} y ε_{m1} (para $\varepsilon_{ch1} = \varepsilon_{m1}$)

Como es de esperar, la BER aumenta al realizar la medida de Bob para $\varepsilon_{m1} = [0,100]\%$ simulando el error introducido por los instrumentos de medida (*single photon detectors*).

En comparación al protocolo BB84, los resultados obtenidos en este escenario son los mismos (Fig. 3.9), ambos protocolos presentan un valor de BER aproximadamente igual para un ε_{m1} dado. Con respecto al protocolo B92 para este mismo escenario (Fig. 3.13), observamos que la BER es más baja para $\varepsilon_{ch1} = \varepsilon_{m1} < 90\%$, a partir de ese punto ambas son aproximadamente igual. Esto es debido, principalmente, a la simplificación del algoritmo del protocolo B92 con respecto los otros dos, provocando que cualquier participación de error externo haga aumentar en gran medida la BER.

A continuación, se realizará una segunda ejecución en la que introducimos a Eva, con las siguientes variables de entrada:

1. Protocolo: EPR.
2. Bytes: 1280 bytes.
3. Comunicación entre Alice y Bob interceptada por Eva (Fig. 3.3).
4. Probabilidad de error de canal ε_{ch1} (Alice y Eva) y ε_{ch2} (Eva y Bob): [0, 100]%.
5. Probabilidad de error de medida ε_{m1} y ε_{m2}: [0, 100]%.

El siguiente gráfico nos muestra la variación de la tasa de error binario (BER) de la *reconciled key* y de la *secret key* en función del error ε_{ch1}, ε_{ch2}, ε_{m1} y ε_{m2} (para $\varepsilon_{ch1} = \varepsilon_{ch2} = \varepsilon_{m1} = \varepsilon_{m2}$) durante una comunicación entre Alice y Bob interceptada por Eva:

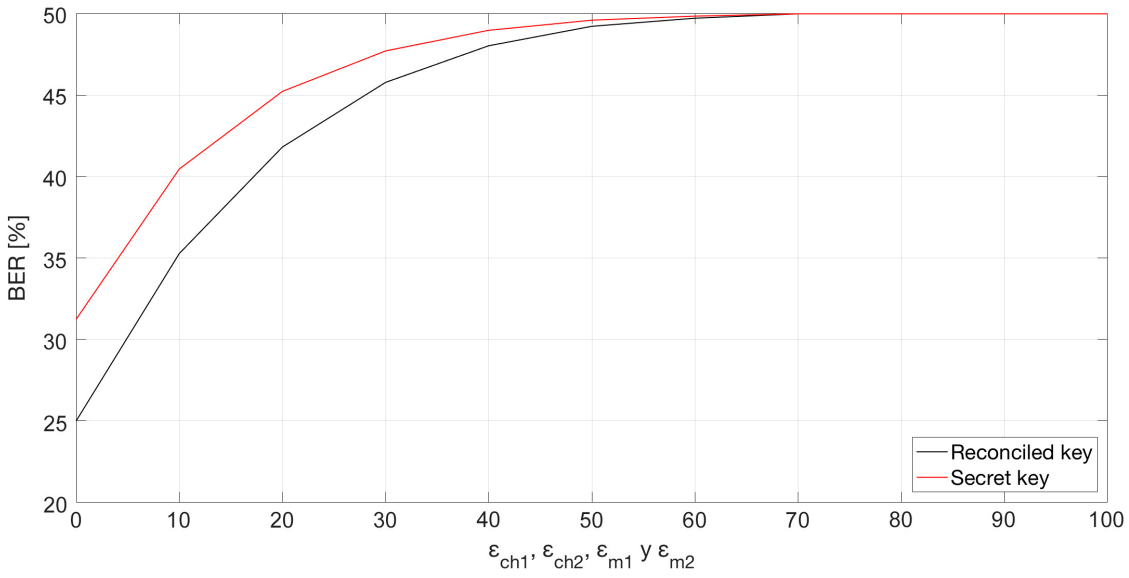

Fig. 3.19 BER de la *reconciled key* y de la *secret key* en función del error ε_{ch1}, ε_{ch2}, ε_{m1} y ε_{m2} (para $\varepsilon_{ch1} = \varepsilon_{ch2} = \varepsilon_{m1} = \varepsilon_{m2}$)

Añadir el error de canal y de medida en la comunicación provoca, como es lógico, para los tres protocolos, un aumento significativo de la BER. Para el conjunto de errores de canal y medida $\varepsilon = 50\%$, los protocolos BB92 y EPR convergen para una $BER = 50\,\%$ aproximadamente, provocando que la comunicación sea totalmente aleatoria (Fig. 3.10). En cambio, para el protocolo B92 con un conjunto de errores de canal y medida $\varepsilon = 0\%$, es decir, solo con la intervención de Eva, tenemos una $BER = 50\%$ de forma constante (Fig. 3.13).

3.5.3. Estudio de la eficiencia

El siguiente gráfico nos muestra la variación de la eficiencia (Eff) para las variaciones estudiadas de probabilidad de error de canal y medida, durante una comunicación directa entre Alice y Bob y con la intervención de Eva:

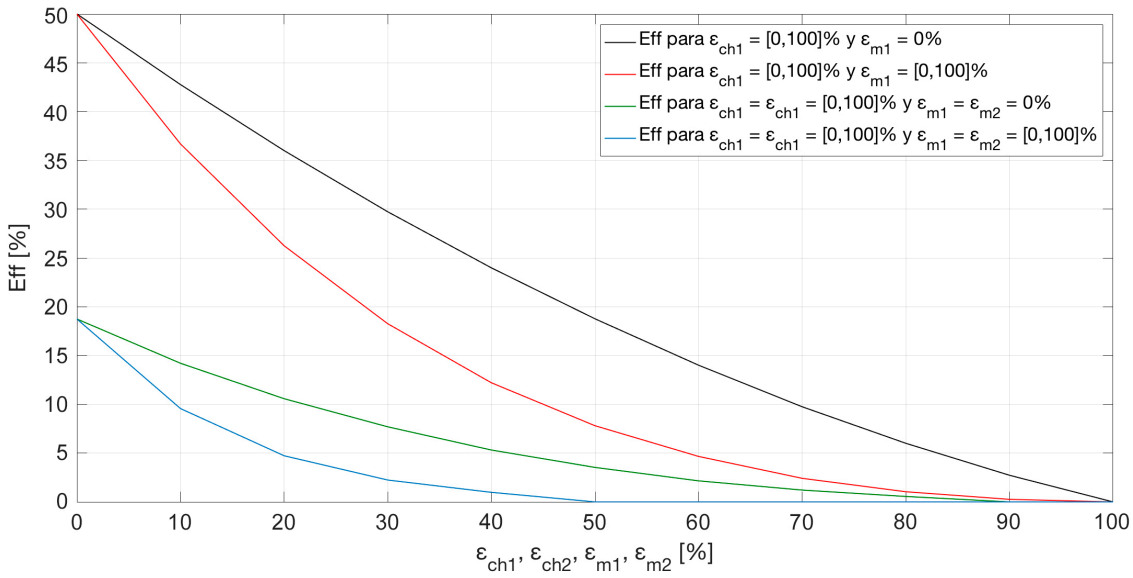

Fig. 3.20 Eficiencia (Eff) del protocolo EPR

El protocolo EPR, durante una comunicación directa entre Alice y Bob para un error de canal y de medida nulo (condiciones ideales), presenta una $Eff = 50\%$, igual que el protocolo BB84 y un 25% más con respecto el B92.

Por otra parte, tras la intervención de Eva tenemos una $Eff = 18.74\%$, es decir, la participación de un intruso durante una sesión de distribución clave cuántica (*QKD*) para el protocolo EPR disminuye la eficiencia (Eff) en, aproximadamente, un 31.26% con respecto una comunicación directa entre el emisor y receptor original.

CAPÍTULO 4. CANALES CUÁNTICOS

En un sistema de comunicación cuántica, la señal está compuesta por estados cuánticos individuales, los cuales no se pueden copiar y repetir de acuerdo al teorema de no clonación. Con la tecnología actual es imposible amplificar o repetir la señal, esto provoca que las comunicaciones tengan una baja eficiencia, es decir, una baja cantidad de fotones que llegan al receptor.

En este capitulo, se realiza el estudio de los dos principales canales cuánticos: la fibra óptica y el espacio libre (ver [11]). El canal cuántico puede ser caracterizado con el objetivo de diseñar una mejor configuración antes del inicio de la sesión *QKD*. Ademas, mediante *MATLAB*, se analizará el comportamiento de las pérdidas introducidas por cada tipo de canal mediante el *script* programado (QC.m), con el objetivo de compararlos en un posible escenario real.

4.1. Enlaces de fibra óptica

El desarrollo de la fibra óptica está determinado por el gran ancho de banda de transmisión, es decir, la posibilidad de enviar una gran cantidad de información por unidad de tiempo. Aunque todo se ve limitado por las grandes pérdidas en función de la distancia.

Los enlaces de fibra óptica presentan pérdidas debido a procesos de dispersión aleatoria y dependen exponencialmente de la longitud. Para describir las propiedades del canal de fibra óptica definimos el coeficiente de perdidas por la distancia α [dB/km] y la longitud l [km] del canal. El parámetro α depende de la longitud de onda (λ) utilizada, para $\lambda = 1330$ nm el coeficiente $\alpha = 0.34$ dB/km, y para $\lambda = 1550$ nm el coeficiente $\alpha = 0.2$ dB/km. El modelo de pérdidas muestra la dependencia con ambos parámetros:

$$L = 10^{-\alpha l/10} \tag{4.1}$$

4.2. Enlaces de espacio libre

Los canales cuánticos ópticos para espacio libre se utilizan principalmente en enlaces de corta distancia terrestres y, de forma muy reciente, en comunicaciones entre satélites a larga distancia. Las pérdidas se pueden dividir en geométricas y atmosféricas.

Las pérdidas atmosféricas vienen definidas por los diferentes fenómenos meteorológicos y sus respectivos coeficientes de atenuación α [dB/km], algunos ejemplos de ellos son:

1. Lluvia: El radio de la gota de agua puede ser mayor que la longitud de onda emitida provocando un efecto atenuador sobre la luz. La reducción de la distancia en el enlace viene dada por la cantidad de lluvia por unidad de tiempo, por ejemplo, para una lluvia de 25 mm/h tenemos un coeficiente $\alpha = 6$ dB/km.

2. Nieve: La atenuación ronda entre los 3 y 30 dB/km debido a que las partículas de la nieve son mucho mayores que las de la lluvia.

3. Niebla: Dependiendo de la densidad de niebla, presenta una atenuación de entre 10 y 100 dB/km.

Para un día despejado, tenemos una atenuación en la señal de $\alpha < 0.1$ dB/km.

Por otro lado, las pérdidas geométricas depende del ancho de haz del transmisor óptico (θ), su longitud (l) y el diámetro de apertura del emisor (d_t) y receptor (d_r). Las pérdidas geométricas dependen principalmente de la divergencia así como de la distancia. El modelo de pérdidas puede determinarse mediante la siguiente ecuación:

$$L_{geo} = (\frac{d_r}{d_t + \theta l})^2 \quad \textbf{(4.2)}$$

Un modelo simple de pérdidas con linea de visibilidad directa (*line of sight*), es decir, sin obstáculos visibles entre emisor y receptor, sería el siguiente:

$$L = L_{geo} 10^{-\alpha l/10} = (\frac{d_r}{d_t + \theta l})^2 10^{-\alpha l/10} \quad \textbf{(4.3)}$$

4.3. Variables de entrada y salida del *script* QC

El *script* programado (QC.m) nos permiten simular, en función de un conjunto de parámetros, el comportamiento de un canal cuántico.

Una vez ejecutado el fichero, la Ventana de Comandos tendrá el siguiente aspecto:

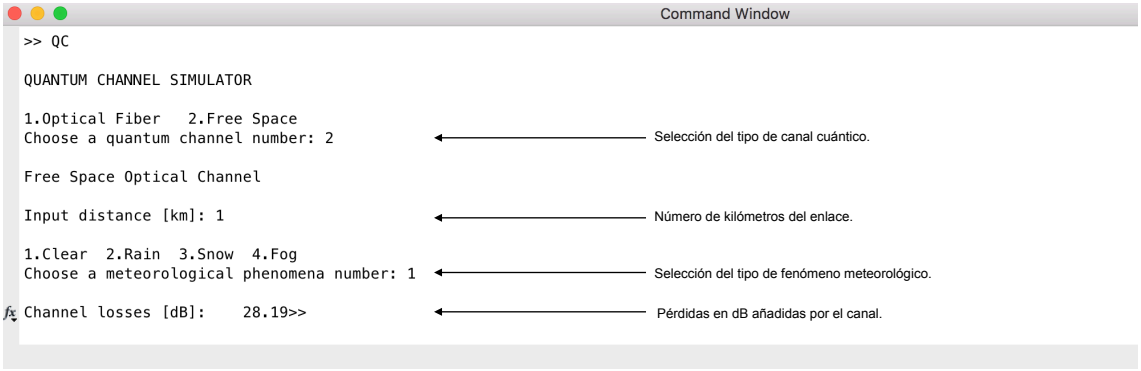

Fig. 4.1 Variables de entrada y salida en la Ventana de Comandos de *MATLAB*

El primer parámetro que tenemos que introducir es el número asociado al modelo de pérdidas del canal cuántico. Para el caso de la fibra óptica se utiliza el modelo de pérdidas descrito en (4.1) y para el espacio libre se utiliza el modelo descrito en (4.3). A continuación, introducimos la distancia en kilómetros que determinarán la longitud del enlace. El último parámetro de entrada nos permite modificar el coeficiente de atenuación α, introduciendo el número asociado al fenómeno meteorológico en el caso del espacio libre. En función de los parámetros introducidos, el *script* calcula las pérdidas en dB introducidas por el canal cuántico seleccionado.

4.4. Cálculo del balance del enlace

A modo de ejemplo se establece un enlace de 8 km de longitud dentro de la isla de Manhattan en New York, con línea de visibilidad directa entre ambos puntos y durante un día soleado.

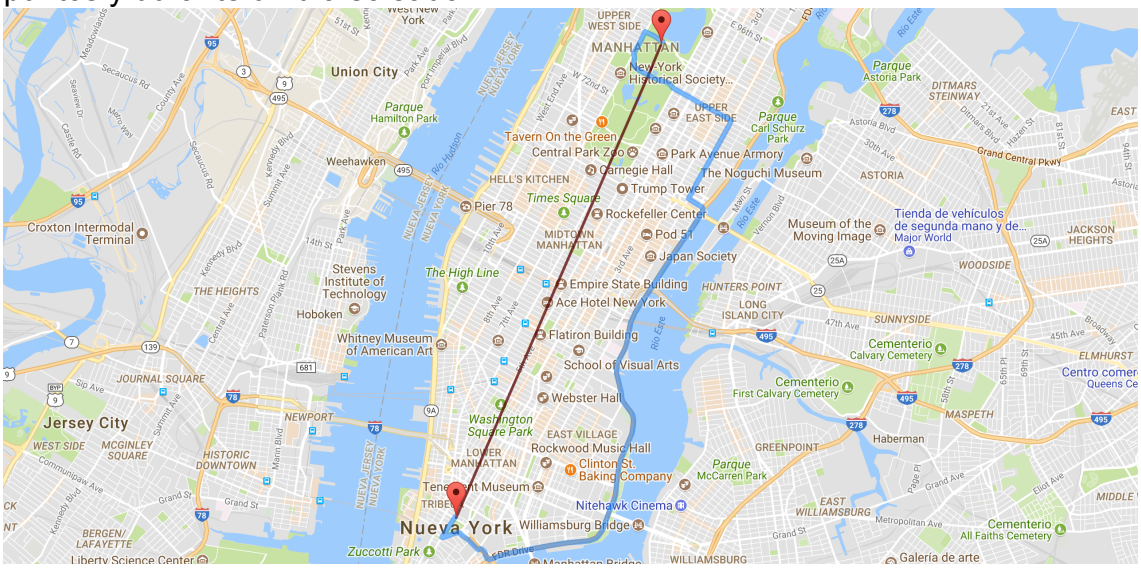

Fig. 4.2 Mapa de distancia del enlace

Para realizar una comparativa empírica entre los enlaces de fibra óptica y de espacio libre, se requiere estudiar el balance del enlace (*link budget*):

$$P_T = P_{rx} - P_{tx} = -\sum L + \sum G \qquad \textbf{(4.4)}$$

Donde P_T es la diferencia en dB de la potencia recibida (P_{rx}) y la potencia transmitida (P_{tx}), con $\sum L$ como la suma de todas las pérdidas y $\sum G$ como la suma de todas las ganancias. Dado que la señal esta compuesta por estados cuánticos individuales que nos se pueden copiar ni repetir, la suma de ganancias es nula, $\sum G = 0$.

4.4.1. Link budget: fibra óptica

Este cálculo estima la pérdida total del enlace a través de un enlace de fibra óptica particular donde se conoce la longitud de la fibra, así como el número de conectores y empalmes. La siguiente tabla incluye los valores de pérdidas comúnmente aceptados.

Tabla 4.1. Valores estandarizados por TIA/EIA utilizados para la transmisión en enlaces de fibra óptica

λ [nm]	α [dB/km]	Pérdidas del connector [dB]	Pérdidas del empalme [dB]
1330	0.34	0.75	0.1
1550	0.20	0.75	0.1

Asumimos un enlace de fibra óptica de 8 km para $\lambda = 1330$ nm con 2 conectores, 4 empalmes y margen de potencia (necesario para compensar al degradación del enlace) de 3 dB:

$$L_T = 0.34[dB/km] \cdot 8[km] + 0.5[dB] \cdot 5 + 0.75[dB] \cdot 2 + 3[dB] = 9.72dB$$

En este ejemplo, se necesitará una potencia estimada superior a 9.72 dB para transmitir a través del enlace.

A continuación, suponemos que el enlace se establece entre New York y Boston, es decir, con una longitud de 300 km.

Como en el anterior ejemplo, asumimos una $\lambda = 1330$ nm con 2 conectores, 4 empalmes y margen de potencia de 3 dB:

$$L_T = 0.34[dB/km] \cdot 300[km] + 0.5[dB] \cdot 5 + 0.75[dB] \cdot 2 + 3[dB] = 109dB$$

En este segundo ejemplo, se necesitará una potencia estimada superior a 109 dB para transmitir a través del enlace.

Los resultados obtenidos son coherentes con respecto a los publicados en diferentes artículos, como por ejemplo el escrito por *IMC Networks* (ver [6]).

4.4.2. Link budget: espacio libre

Este cálculo estima la pérdida total del enlace a través de un enlace de espacio libre particular donde se conoce la longitud del enlace, así como las dimensiones físicas del diámetro de apertura del emisor (d_t) y receptor (d_r) y el ángulo de divergencia del haz (θ).

En el siguiente ejemplo, se utilizan los valores típicos para un sistema óptico de espacio libre (ver [12]). Las pérdidas geométricas descritas en (4.2) para $d_t = 3$ cm, $d_r = 8$ cm y $\theta = 2$ mrad serán:

$$L_{geo}[dB] = -20log(\frac{d_r}{d_t + \theta l}) = -20log(\frac{8 \cdot 10^{-2}[m]}{3 \cdot 10^{-2}[m] + 2[mrad] \cdot 8[km]}) = 46.04dB$$

Las pérdidas totales descritas en (4.3), asumiendo un día despejado con una atenuación en la señal de $\alpha < 0.1$ dB/km serán:

$$L_T[dB] = -20log(\frac{d_r}{d_t + \theta l}) - 10log(10^{\alpha l/10}) = 46.04 + 0.8 = 46.84dB$$

En este ejemplo, se necesitará una potencia estimada superior a 46.84 dB para transmitir a través del enlace.

Los resultados obtenidos son coherentes con respecto a los presentados en el libro escrito por Heinz Willebrand (ver [12]).

A continuación, suponemos que el enlace se establece entre New York y Boston, es decir, con una longitud de 300 km.

Las pérdidas geométricas descritas en (4.2) para $d_t = 3$ cm, $d_r = 8$ cm y $\theta = 2$ mrad serán:

$$L_{geo}[dB] = -20log(\frac{d_r}{d_t + \theta l}) = -20log(\frac{8 \cdot 10^{-2}[m]}{3 \cdot 10^{-2}[m] + 2[mrad] \cdot 300[km]}) = 77.5dB$$

Las pérdidas totales descritas en (4.3), asumiendo un día despejado con una atenuación en la señal de $\alpha < 0.1$ dB/km serán:

$$L_T[dB] = -(20log(\frac{d_r}{d_t + \theta l}) - 10log(10^{\alpha l/10})) = 77.5 + 30 = 107.5dB$$

En este segundo ejemplo, se necesitará una potencia estimada superior a 109 dB para transmitir a través del enlace.

4.5. Simulación y análisis del canal cuántico

Haciendo uso del *script* (QC.m), se analizará la evolución de las pérdidas de canal en función de la distancia.

El primer estudio se realizará sobre un enlace de corta distancia (1-10 km) con el uso de fibra óptica.

El gráfico 4.3 nos muestra las pérdidas únicamente añadidas por el modelo de pérdidas de fibra óptica definido en (4.1) en función de la distancia para $\lambda = 1330$ y $\lambda = 1550$ nm:

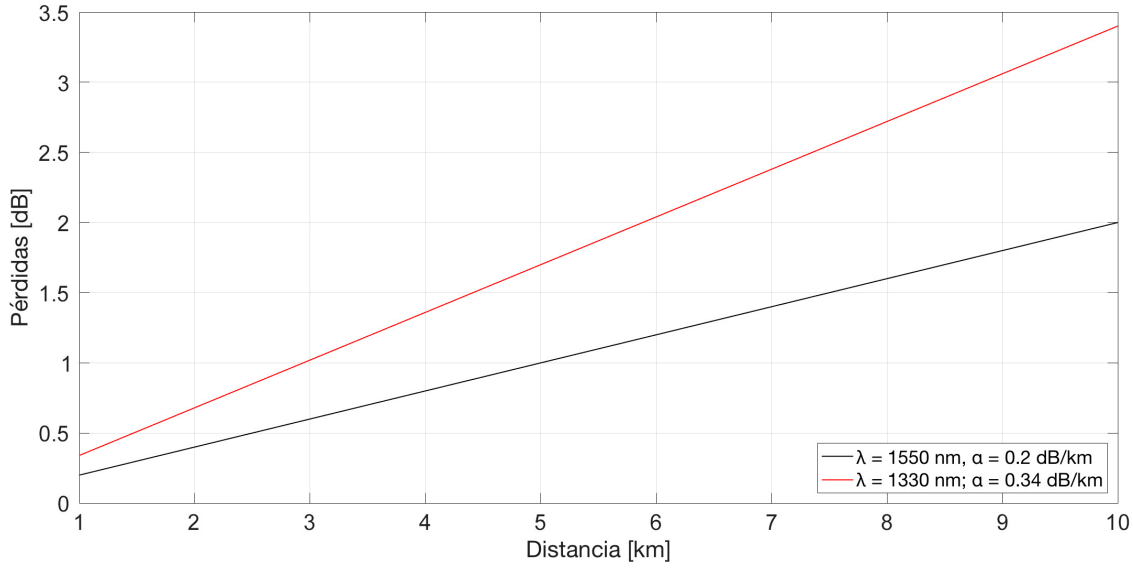

Fig. 4.3 Pérdidas de canal en función de la distancia en fibra óptica

Siguiendo la ecuación descrita en (4.1) se observan unas pérdidas (L [dB]) que crecen de forma lineal con la distancia para $\lambda = 1550$ y $\lambda = 1330$ nm.

La frecuencia (f) tiene una relación inversa con la longitud de onda (λ), a mayor frecuencia menor longitud de onda y viceversa. Por tanto, se concluye que para una mayor frecuencia tenemos una mayor α, es decir, más atenuación en función de la distancia.

El segundo estudio se realizará sobre un enlace de corta distancia (1-10 km) con el uso del espacio libre.

El gráfico 4.4 nos muestra las pérdidas del modelo de espacio libre definido en (4.3) en función de la distancia para diferentes valores de α:

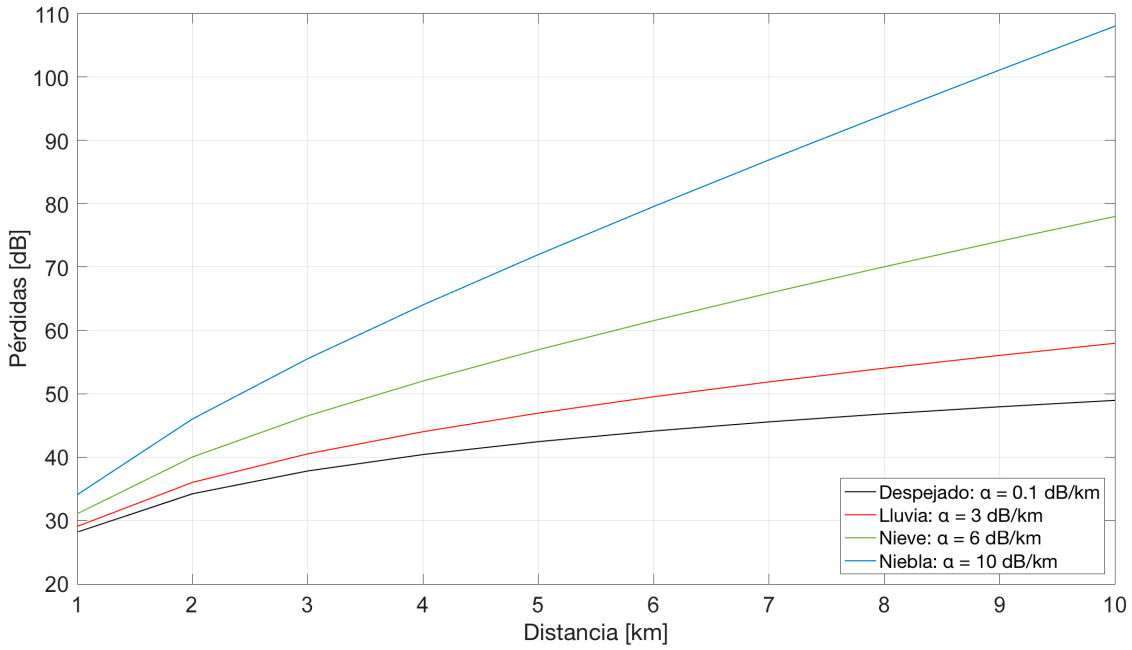

Fig. 4.4 Pérdidas de canal en función de la distancia en espacio libre

En función del fenómeno meteorológico que afecte a la comunicación del enlace se obtiene un incremento de las pérdidas (L [dB]) en función de la distancia. Esto provoca que este tipo de comunicaciones sean complejas de realizar bajo condiciones climatológicas poco favorables.

Es importante aclarar que los valores de atenuación por distancia utilizados para cada uno de los diferentes fenómenos meteorológicos, se basan en una aproximación. Estos valores dependen de la cantidad de lluvia por unidad de tiempo, el grosor de las partículas de la nieve y la densidad de la niebla.

El tercer y último estudio se realiza sobre un enlace de larga distancia (50-500 km) con el uso de fibra óptica y del espacio libre con el fin de realizar una comparativa entre ambos.

El gráfico 4.5 nos muestra las pérdidas del modelo de fibra óptica (4.2) para un valor de $\lambda = 1330$ nm y del modelo de espacio libre (4.3) para un valor de $\alpha = 0.1$ dB/km (día despejado):

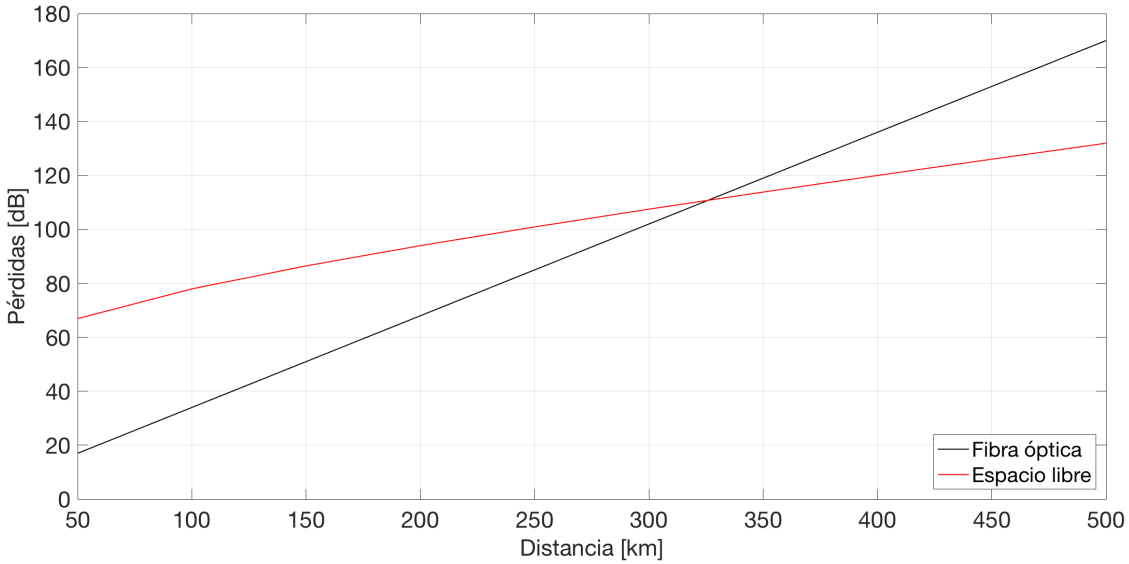

Fig. 4.5 Pérdidas de canal en función de la distancia en fibra óptica y espacio libre

Los resultados obtenidos nos muestran que para enlaces de larga distancia, la fibra óptica presenta más pérdidas que el espacio libre.

Los enlaces de espacio libre permiten un rango máximo para enlaces terrestres del orden de 2.3 km (ver [3]), pero la estabilidad del enlace es muy dependiente de los factores atmosféricos como la lluvia, la nieve y la niebla. Por ese motivo, es mas recomendable utilizar enlaces de fibra óptica para realizar comunicaciones de corta distancia.

Por otra parte, una solución a la comunicación entre enlaces de larga distancia, es el uso de satélites. Estos enlaces permiten alcanzar en la actualidad una distancia del orden de varios miles de kilómetros (ver [2]).

4.6. Estimación del error de canal

Este último apartado se plantea como una introducción a una posible ampliación del estudio tratado en esta memoria.

El análisis realizado en relación a los canales cuánticos, trae como consecuencia, la posibilidad de relacionar las pérdidas introducidas por los enlaces de fibra óptica y espacio libre con el error de canal (ε_{ch}), definido en el capitulo 3, aplicado sobre los diferentes protocolos *QKD*.

Tal y como está definido el código del *script* QKD, se entiende como error de canal, el porcentaje de fotones que serán recibidos por Bob de forma aleatoria con respecto el fotón original enviado por Alice.

La principal variación a la que se enfrenta esta ampliación, se basa en la modificación del código del *script* QKD de tal forma que tras añadir un error de canal, Bob sólo reciba el porcentaje de fotones restantes (para $\varepsilon_{ch} = 2\%$, Bob recibe el 98% de los fotones). La aleatoriedad, con respecto el fotón original enviado por Alice, se aplicará en función del porcentaje del error de medida (ε_{m}).

El porcentaje de fotones recibidos vendrá dado por el cociente entre la potencia recibida (P_{rx}) y la potencia transmitida (P_{tx}) utilizadas en el balance del enlace (*link budget*) del canal cuántico (fibra óptica o espacio libre). De esta forma, los *scripts* QKD (*Quantum Key Distribution*) y QC (*Quantum Channel*) podrán coexistir añadiendo la posibilidad al usuario de simular un escenario más realista.

CONCLUSIONES

El objetivo principal de este trabajo ha sido analizar el comportamiento de los protocolos *QKD* BB84, B92 y EPR, para diferentes escenarios definidos en función de los parámetros de entrada insertados en el *script* QKD. A continuación, se realiza la comparativa de los resultados obtenidos en la simulación de cada escenario.

El primer estudio concluye que para un número de 1280 bytes, el valor de la BER obtenido en las simulaciones presenta un error a partir del tercer decimal en comparación al valor calculado teóricamente, por tanto, para el resto de simulaciones se utiliza este valor con el objetivo de obtener resultados representativos sin la necesidad de aumentar el coste computacional de cada simulación. Además, se analiza la diferencia del valor de la BER para 1280 bytes en el caso de la *reconciled* key y la *secret key*, obteniendo como resultado que tras la aplicación del operador lógico XOR, aumenta la complejidad de la clave y, por tanto, para un mismo porcentaje de error se obtiene una BER más elevada en el caso de la *secret key*.

A continuación, se analiza el error introducido por el canal en el caso donde Alice y Bob realizan una comunicación directa (sin intruso). En este escenario se concluye que para los tres protocolos la BER de la *reconciled key* viene dada por $BER \simeq \varepsilon_{ch1}/2$, donde ε_{ch1} es la probabilidad de error de canal. En cambio, la BER de la *secret key* es mayor para el B92 respecto el BB84 y el EPR. Esto es consecuencia del número de bits útiles utilizados para generar la *reconciled key*. Por tanto, el protocolo B92 es mas sensible al error al aplicarle un mecanismo de ampliación de privacidad.

El siguiente escenario, añade el error de medida durante una comunicación directa entre Alice y Bob. Los resultados obtenidos nos indican que, el error introducido por los instrumentos de medida (*single photon detectors*), provocan un aumento en la BER, tanto de la *reconciled key* como de la *secret key*, con respecto el escenario anterior (únicamente error de canal). De nuevo, el protocolo B92 presenta una mayor sensibilidad entre el 0 y el 90% de error (canal y medida), a partir de ese punto, los tres protocolos son aproximadamente iguales. Esto es debido a que el algoritmo es más simple (únicamente utiliza dos tipos de fotones polarizados: $|0\rangle$ y $|+\rangle$) provocando que cualquier participación de error externo haga aumentar en gran medida la BER.

Tras este análisis inicial, se modificó el esquema de la comunicación con la intención de estudiar la alteración de los resultados durante la participación de un intruso entre Alice y Bob. Los siguientes resultados demuestran que sin la necesidad de añadir ningún error de canal o medida, los tres protocolos son capaces de detectar la participación de Eva ($BER > 0\%$ sin errores externos de canal y medida):

El protocolo BB84 presenta una $BER \simeq 37.5\%$ para la *reconciled key* y una $BER \simeq 42.5\%$ para la *secret key*, para el B92 el valor aumenta hasta $BER \simeq 50\%$, tanto para la *reconciled key* como para la *secret key*. Por último, el EPR muestra un valor de $BER \simeq 25\%$ y $BER \simeq 31.5$ para la *reconciled key* y *secret key*, respectivamente. Según los resultados obtenidos, se concluye que Eva no puede realizar la medida en el canal cuántico sin ser detectada.

En condiciones de entorno ideal (error de canal y medida nulo), el protocolo que presenta un mayor umbral de detección a la intervención de Eva es el B92, aunque hay que tener en cuenta que también es el protocolo más sensible a los errores externos (canal y medida). Por otra parte, en un escenario más realista, donde tanto el error de canal como el de medida no son despreciables, es preferible el uso del protocolo BB84 o el EPR. Esto es debido a que son más robustos contra cualquier tipo de error externo, permitiendo detectar de forma más sencilla el error introducido por Eva.

Esta última afirmación se deduce a partir de los resultados obtenidos para el escenario anterior (únicamente intervención de Eva) en comparación al escenario donde participa tanto Eva como los errores externos. En ese estudio, sólo se identifica un aumento de BER entre ambos escenarios para los protocolos BB84 y EPR. Para el B92, en cambio, se presenta una $BER \simeq 50\%$ en ambos escenarios.

A nivel de eficiencia, se obtiene como resultado que tanto el BB84 como el EPR tienen un 50% en condiciones ideales, en cambio, el protocolo B92 presenta una eficiencia del 25%. Esto es debido a que únicamente el 25% de los bits de la secuencia post medida, son utilizados para generar la *reconciled key* (bits útiles). Además, el valor de la eficiencia tras la intervención de Eva disminuye en todos los protocolos, siendo el B92 con una $Eff = 0\%$ el más afectado, seguido por el BB84 con una $Eff = 7.82\%$ y, por último el EPR con una $Eff = 18.74\%$.

En el último capitulo, se establece el estudio de los modelos matemáticos estudiados para el cálculo del error introducido por el canal cuántico, donde se concluye de manera teórica y práctica que los canales de fibra óptica presentan un menor número de pérdidas para enlaces de corta distancia en comparación a los de espacio libre, pero, si recurrimos a enlaces de larga distancia, las pérdidas de los enlaces de fibra llegan a superar a los de espacio libre. Además, el papel de los fenómenos meteorológicos afecta a la estabilidad de los enlaces de espacio libre provocando una dependencia climatológica.

BIBLIOGRAFÍA

[1] Desurvire, E., *Classical and quantum information theory. An introduction for the telecom scientist*, Cambridge, Cambridge University Press (2009).

[2] European Space Agency., “Another world first for Artemis: a laser link with an aircraft”, *United Space in Europe*, European Space Agency (2006).

[3] Garlington, T., Babbitt, J. and Long, G., “Analysis of Free Space Optics as a Transmission Technology”, *Sams*, 1, 1-12 (2005).

[4] Gilat, A., *Matlab. Una introducción con ejemplos prácticos*, Reverté, Barcelona (2006).

[5] Gisin, N., *et al. “*Quantum cryptography*”, Reviews of modern physics*, 74, 135-195 (2002).

[6] IMC Networks., *Calculating Fiber Loss and Distance*, IMC Networks, United States (2009).

[7] Nielsen, M.A. and Chuang, I.L., *Quantum Computation and Quantum Information*, Cambridge University Press, New York (2010).

[8] Nordholt, J.E and Hughes, R-J., “A new face for cryptography”, *Los Alamos Science*, 27, 69-85 (2002).

[9] Qiao, H. and Chen, X.Y., “Simulation of BB84 Quantum Key Distribution in depolarizing channel”, *Scientific Research, 483-487(2009).*

[10] Scarani, V., “Una bonita idea”, Cap. 5 en *Física cuántica interferencias, correlaciones y realidad*, Iniciativa digital politècnica, pp. 59-67, Barcelona (2015).

[11] Scarani, V., Bechmann-Pasquinucci, H., Cerf, N-J., Dusek, M., Lutkenhaus, N and Peev, M., “The security of practical quantum key distribution”, *Reviews of Modern P*hysics, 81 (3), 1-52 (2009).

[12] Willebrand, H. and Ghuman, B.S., “Fundamentals of FSO Technology”, Cap. 2 en *Free Space Optics: Enabling Optical Connectivity in Today’s Networks*, Sams Publishing, pp.43-44, United States (2001).

ANEXO

A.1. Ejemplo de sesión de distribución de clave cuántica: protocolo B92

El protocolo B92 empieza con Alice generando una secuencia aleatoria de bits clásicos (cbits). Los cbits de ésta secuencia los llamamos a_k y podrían ser los siguientes:

$$00101101$$

En función del valor de a_k, Alice crea un fotón (por tanto, un qubit) $|0\rangle$ o $|+\rangle$ (Tabla 2.6). Por tanto, en este ejemplo, el estado cuántico que tendrá Alice y que enviará a Bob sería el siguiente:

$$|0\rangle\,|0\rangle\,|+\rangle\,|0\rangle\,|+\rangle\,|+\rangle\,|0\rangle\,|+\rangle$$

Bob recibe este estado cuántico y ha de decidir en que base mide cada quibit. Por eso genera otra secuencia aleatoria de cbits que llamamos b_k. Por ejemplo:

$$10101011$$

La base que utilizará será la Z o la X (Tabla 2.2). Por tanto, con esta secuencia en particular, las bases de medida serán:

$$XZXZXZXX$$

Ahora realiza la medida de todos los qubits según esta elección de bases, obtendrá los cbit asociados a las bases Z y X obteniendo la secuencia que llamaremos b'_k. De esta manera, la secuencia b'_k que obtenemos es la siguiente:

$$\frac{0}{1}0000\frac{0}{1}\frac{0}{1}0$$

Donde 0/1 indica que podemos tener un 0 o un 1 con una probabilidad del 50% cada uno. Por tanto, una posible secuencia sería la siguiente:

$$10000100$$

Y es esta secuencia de b'_k la que Bob envía a Alice por un canal público. Alice se quedará con aquellos cbits de la secuencia a_k que correspondan a las posiciones donde $b'_k = 1$, generando una secuencia (*reconciled key*) de cbits que llamamos a'_k:

$$A_{reconciled-key} = 01$$

Mientras que Bob negará los cbits de la secuencia b_k que corresponden a las posiciones donde $b'_k = 1$ y genera su propia *reconciled key* de cbits que llamamos b''_k**:**

$$B_{reconciled-key} = \overline{10} = 01$$

A continuación, Alice y Bob generan una nueva clave (*secret key*) con un mayor nivel de seguridad (ampliación de privacidad) a partir de la *reconciled key*. Para realizar esta operación, se utiliza tal y como hemos comentado anteriormente el operador lógico XOR ($\oplus$). En este ejemplo, Alice y Bob generan la siguiente *secret key* de 8 cbits de longitud:

$$a''_3 = a''_1 \oplus a''_2 = 0 \oplus 1 = 1 \qquad b''_3 = b''_1 \oplus b''_2 = 0 \oplus 1 = 1$$
$$a''_4 = a''_2 \oplus a''_3 = 1 \oplus 1 = 0 \qquad b''_4 = b''_2 \oplus b''_3 = 1 \oplus 1 = 0$$
$$a''_5 = a''_3 \oplus a''_4 = 1 \oplus 0 = 1 \qquad b''_5 = b''_3 \oplus b''_4 = 1 \oplus 0 = 1$$
$$a''_6 = a''_4 \oplus a''_5 = 0 \oplus 1 = 1 \qquad b''_6 = b''_4 \oplus b''_5 = 0 \oplus 1 = 1$$
$$a''_7 = a''_5 \oplus a''_6 = 1 \oplus 1 = 0 \qquad b''_7 = b''_5 \oplus b''_6 = 1 \oplus 1 = 0$$
$$a''_8 = a''_6 \oplus a''_7 = 1 \oplus 0 = 1 \qquad b''_8 = b''_6 \oplus b''_7 = 1 \oplus 0 = 1$$

$$A_{secret-key} = 01101101$$
$$B_{secret-key} = 01101101$$

Como se trata de un ejemplo teórico con condiciones ideales, tanto Alice como Bob tienen la misma clave y, por tanto, la tasa de error binario (BER) es del 0%.

Resumiendo todos los pasos en formato tabla donde la zona de color corresponde a la parte de Alice y la blanca a la parte de Bob (en rojo se indica

todos los números que son resultado de un proceso aleatorio) se concluye el resultado mostrado en la tabla A.1.1.

Tabla A.1.1. Resumen del desarrollo práctico para el protocolo B92

a_k	0	0	1	0	1	1	0	1
Qubit	$\|0\rangle$	$\|0\rangle$	$\|+\rangle$	$\|0\rangle$	$\|+\rangle$	$\|+\rangle$	$\|0\rangle$	$\|+\rangle$
b_k	1	0	1	0	1	0	1	1
Base	X	Z	X	Z	X	Z	X	X
Medida	± 1	$+1$	$+1$	$+1$	$+1$	± 1	± 1	$+1$
Posibles b'_k	0/1	0	0	0	0	0/1	0/1	0
b'_k	1	0	0	0	0	0	1	1
$A_{secret-key}$	0	1	1	0	1	1	0	1
$B_{secret-key}$	0	1	1	0	1	1	0	1

Por último, obtenemos la eficiencia (Eff) para este ejemplo utilizando la ecuación descrita en (2.3):

$$Eff = \frac{n_{rk} - n_{error}}{n_{bits}} = \frac{2}{8} = 0.25$$

Un 25% de los qubits generados inicialmente son utilizados para elaborar la *reconciled key* de esta comunicación.

A.2. Ejemplo de sesión de distribución de clave cuántica: protocolo EPR

El protocolo EPR empieza generando un conjunto de pares EPR (compartidos por Alice y Bob), basados en uno de los cuatro estados de Bell, por ejemplo:

$$\left|\beta_{00}\right\rangle = \frac{1}{\sqrt{2}}(\left|00\right\rangle + \left|11\right\rangle) = \frac{1}{\sqrt{2}}(\left|++\right\rangle + \left|--\right\rangle)$$

Alice y Bob generan dos secuencias aleatorias de bits clásicos (cbits). Los cbits de estas secuencias los llamamos a_k y b_k. Podrían ser, respectivamente, los siguientes:

$$00010001$$
$$00100011$$

Las bases que utilizarán será la Z o la X (Tabla 2.2). Por tanto, con estas secuencias en particular, las bases de medida serán:

$$ZZZXZZZX$$
$$ZZXZZZXX$$

Alice realiza la medida sobre la superposición de estados de $|0\rangle$ y $|1\rangle$ o $|+\rangle$ y $|-\rangle$ según su elección de bases, provocando un colapso instantáneo y obteniendo, de manera totalmente aleatoria, los cbits asociados a las bases Z y X. Así pues, un posible resultado de la medida que obtiene Alice sería:

$$01111001$$

El estado cuántico resultante después de la medida de Alice es el siguiente:

$$|0\rangle\ |1\rangle\ |1\rangle\ |-\rangle\ |1\rangle\ |0\rangle\ |0\rangle\ |-\rangle$$

Bob realiza la medida sobre la pareja del fotón entrelazado que comparte con Alice y que ya ha colapsado en una de las dos posibilidades. Según su elección de bases, obtendrá los cbit asociados a las bases Z y X obteniendo la secuencia que llamaremos b'_k. De esta manera, la secuencia b'_k que obtenemos es la siguiente:

$$01\frac{0}{1}\frac{0}{1}10\frac{0}{1}1$$

Donde 0/1 indica que podemos tener un 0 o un 1 con una probabilidad del 50% cada uno. Por tanto, una posible secuencia sería la siguiente:

$$01001001$$

Alice envía la secuencia a_k a Bob por un canal público y Bob responde con las posiciones donde $a_k = b_k$. De esta manera, las posiciones serían:

$$1^a,\ 2^a,\ 5^a,\ 6^a \text{ y } 8^a$$

Alice se quedará con aquellos cbits de la secuencia a'_k que correspondan a las posiciones que Bob ha enviado, generando una secuencia (*reconciled key*) de cbits que llamamos a''_k:

$$A_{reconciled-key} = 01101$$

Mientras que Bob, a partir de los cbits de la secuencia b'_k que corresponden a las posiciones donde $a_k = b_k$, genera su propia *reconciled key* de cbits que llamamos b''_k:

$$B_{reconciled-key} = 01101$$

A continuación, Alice y Bob generan una nueva clave (*secret key*) con un mayor nivel de seguridad (ampliación de privacidad) a partir de la *reconciled key*. Para realizar esta operación, se utiliza tal y como hemos comentado anteriormente el operador lógico XOR ($\oplus$). En este ejemplo, Alice y Bob generan la siguiente *secret key* de 8 cbits de longitud:

$$a''_6 = a''_1 \oplus a''_2 = 0 \oplus 1 = 1 \qquad b''_6 = b''_1 \oplus b'_2 = 0 \oplus 1 = 1$$
$$a''_7 = a''_2 \oplus a''_3 = 1 \oplus 1 = 0 \qquad b''_7 = b''_2 \oplus b''_3 = 1 \oplus 1 = 0$$
$$a''_8 = a''_3 \oplus a''_4 = 1 \oplus 0 = 1 \qquad b''_8 = b''_3 \oplus b''_4 = 1 \oplus 0 = 1$$

$$A_{secret-key} = 01101101$$
$$B_{secret-key} = 01101101$$

Como se trata de un ejemplo teórico con condiciones ideales, tanto Alice como Bob tienen la misma clave y, por tanto, la tasa de error binario (BER) es del 0%.

Resumiendo todos los pasos en formato tabla donde la zona de color corresponde a la parte de Alice y la blanca a la parte de Bob (en rojo se indica todos los números que son resultado de un proceso aleatorio) se concluye el resultado mostrado en la tabla A.2.1.

Tabla A.2.1. Resumen del desarrollo práctico para el protocolo EPR

a_k	0	0	0	1	0	0	0	1
Base	Z	Z	Z	X	Z	Z	Z	X
Medida	$+1$	-1	-1	-1	-1	$+1$	$+1$	-1
a'_k	0	1	1	1	1	0	0	1
Qubit	$\|0\rangle$	$\|1\rangle$	$\|1\rangle$	$\|-\rangle$	$\|1\rangle$	$\|0\rangle$	$\|0\rangle$	$\|-\rangle$
b_k	0	0	1	0	0	0	1	1
Base	Z	Z	X	Z	Z	Z	X	X
Medida	$+1$	-1	± 1	± 1	-1	$+1$	± 1	-1
Posibles b'_k	0	1	0/1	0/1	1	0	0/1	1
b'_k	0	1	0	0	1	0	0	1
$A_{secret-key}$	0	1	1	0	1	1	0	1
$B_{secret-key}$	0	1	1	0	1	1	0	1

Por último, obtenemos la eficiencia (Eff) para este ejemplo utilizando la ecuación descrita en (2.3):

$$Eff = \frac{n_{rk} - n_{error}}{n_{bits}} = \frac{5}{8} = 0.625$$

Un 62.5% de los qubits generados inicialmente son utilizados para elaborar la *reconciled key* de esta comunicación.

A.3. Demostración

$$\left|\beta_{00}\right\rangle = \frac{1}{\sqrt{2}}(\left|00\right\rangle + \left|11\right\rangle) = \frac{1}{\sqrt{2}}(\left|++\right\rangle + \left|--\right\rangle)$$

Dado que $\left|+\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle)$ y $\left|-\right\rangle = \frac{1}{\sqrt{2}}(\left|0\right\rangle - \left|1\right\rangle)$

$$\frac{1}{\sqrt{2}}(\left|++\right\rangle + \left|--\right\rangle) = \frac{1}{\sqrt{2}}[\frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle)\frac{1}{\sqrt{2}}(\left|0\right\rangle + \left|1\right\rangle) + \frac{1}{\sqrt{2}}(\left|0\right\rangle - \left|1\right\rangle)\frac{1}{\sqrt{2}}(\left|0\right\rangle - \left|1\right\rangle)] =$$

$$\frac{1}{\sqrt{2}}[\frac{1}{2}(\left|00\right\rangle + \left|01\right\rangle + \left|10\right\rangle + \left|11\right\rangle + \left|00\right\rangle - \left|01\right\rangle + \left|10\right\rangle - \left|11\right\rangle)] = \frac{1}{\sqrt{2}}[\frac{1}{2}(2\left|00\right\rangle + 2\left|11\right\rangle)] =$$

$$\frac{1}{\sqrt{2}}(\left|00\right\rangle + \left|11\right\rangle)$$

A.4. Código Matlab “Quantum key distribution simulator”

Script QKD.m

```
fprintf('\nQUANTUM KEY DISTRIBUTION SIMULATOR\n')
fprintf('\n1.BB84   2.B92   3.EPR')
n_protocol = input('\nChoose a protocol number: ');

%Protocolo BB84
if n_protocol == 1
   fprintf('\nBB84 PROTOCOL\n')

   n_bytes = input('\nInput bytes: ');
   n_bits = n_bytes*8;
   if n_bits<=0
      fprintf('\nThe input bytes must be real number greater than zero')
      return;
   end
   prompt = ('\nMan in the middle [Y/N]: ');
   mitm = input(prompt, 's');
   if mitm == 'Y'
      fprintf('\nALICE ----- (Channel 1) ----- EVE ----- (Channel 2) ----- BOB\n');
      prob_channel1 = input('Error prob. in 1st channel [%]: ');
      prob_channel1 = prob_channel1/100;
      prob_measure1 = input('Error prob. in Eve measure [%]: ');
      prob_measure1 = prob_measure1/100;
      prob_channel2 = input('Error prob. in 2nd channel [%]: ');
      prob_channel2 = prob_channel2/100;
      prob_measure2 = input('Error prob. in Bob measure [%]: ');
      prob_measure2 = prob_measure2/100;
       if prob_channel1<0 || prob_channel1>1 || prob_channel2<0 || prob_channel2>1
|| prob_measure1<0 || prob_measure1>1 || prob_measure2<0 || prob_measure2>1
                fprintf('\nThe probability must be real number between zero and one
hundred')
          return;
      end
   elseif mitm == 'N'
      prob_measure1 = 0;
      prob_channel2 = 0;
      fprintf('\nALICE ----- (Channel) ----- BOB\n');
      prob_channel1 = input('Error prob. in the channel [%]: ');
      prob_channel1 = prob_channel1/100;
      prob_measure2 = input('Error prob. in Bob measure [%]: ');
      prob_measure2 = prob_measure2/100;
```

```
                    if prob_channel1<0 || prob_channel1>1 || prob_measure2<0 ||
prob_measure2>1
                fprintf('\nThe probability must be real number between zero and one
hundred')
          return;
       end
    else
       fprintf('\nThe input letter must be Y or N')
       return;
    end

    % Alice genera dos vectores de bits clasicos aleatorios: A1, A2
    A1 = rand(1, n_bits) > 0.5; A2 = rand(1, n_bits) > 0.5;

    % Bob genera un vector de bits clasicos aleatorio: B1
    B1 = rand(1, n_bits) > 0.5;

    % Eva genera un vector de bits clasicos aleatorio: E
    E = rand(1, n_bits) > 0.5;

    % Alice genera un vector de bits clasicos resultante: A_res
    A_res = rand(1, n_bits);
    i=0;
    while i<n_bits
       if A1(n_bits-i)==0
          if A2(n_bits-i)==0
             A_res(n_bits-i)=0;% Horizontal - |0>
             i=i+1;
          else
             A_res(n_bits-i)=1;% Vertical - |1>
             i=i+1;
          end
       else
          if A2(n_bits-i)==0
             A_res(n_bits-i)=2;% Right - |+>
             i=i+1;
          else
             A_res(n_bits-i)=3;% Left - |->
             i=i+1;
          end
       end
    end
```

```
% Error de transmision de datos: Canal 1 (Alice - Eva)
  i=0;
  while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if prob_rand <= prob_channel1
      A_res(n_bits-i) = randi([0,3]);
      i=i+1;
    else
      i=i+1;
    end
  end

  % Eva genera un vector de bits clasicos resultante: E_res
  E_res = rand(1, n_bits);
  if mitm == 'Y'
    i=0;
    while i<n_bits
      prob_rand = 0 + (1-0).*rand(1,1);
      if E(n_bits-i)==0 && (A_res(n_bits-i)==0 || A_res(n_bits-i)==1)
        if A_res(n_bits-i)==0
          if prob_rand >= prob_measure1
            E_res(n_bits-i)=0;% Horizontal - |0>
          end
        else
          if prob_rand >= prob_measure1
            E_res(n_bits-i)=1;% Vertical - |1>
          end
        end
        i=i+1;
      elseif E(n_bits-i)==1 && (A_res(n_bits-i)==2 || A_res(n_bits-i)==3)
        if E_res(n_bits-i)==2
          if prob_rand >= prob_measure1
            E_res(n_bits-i)=2;% Right - |+>
          end
        else
          if prob_rand >= prob_measure1
            E_res(n_bits-i)=3;% Left - |->
          end
        end
        i=i+1;
      else
```

```
            E_res(n_bits-i) = randi([0,3]);
            i=i+1;
        end
    end
end

% Error de medida de datos: Medida 1 (Eva)
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if prob_rand <= prob_measure1
        E_res(n_bits-i) = randi([0,3]);
        i=i+1;
    else
        if mitm == 'Y'
            i=i+1;
        else
            E_res(n_bits-i) = A_res(n_bits-i);
            i=i+1;
        end
    end
end

% Error de transmision de datos: Canal 2 (Eva - Bob)
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if prob_rand <= prob_channel2
        E_res(n_bits-i) = randi([0,3]);
        i=i+1;
    else
        i=i+1;
    end
end

% Bob genera un vector de bits clasicos: B2
B2 = rand(1, n_bits) > 0.5;
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if B1(n_bits-i)==0 && (E_res(n_bits-i)==0 || E_res(n_bits-i)==1)
        if E_res(n_bits-i)==0
```

```
% Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
               B2(n_bits-i)=0;
            end
         else
            % Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
               B2(n_bits-i)=1;
            end
         end
         i=i+1;
      elseif B1(n_bits-i)==1 && (E_res(n_bits-i)==2 || E_res(n_bits-i)==3)
         if E_res(n_bits-i)==2
            % Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
               B2(n_bits-i)=0;
            end
         else
            % Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
               B2(n_bits-i)=1;
            end
         end
         i=i+1;
      else
         i=i+1;
      end
   end

   % Deteccion de errores: Bit de paridad
   ones_A1 = find(A1==1); ones_A2 = find(A2==1);
   ones_B1 = find(B1==1); ones_B2 = find(B2==1);
   if mod(length(ones_A1),2)==0
      A1(n_bits+1)=1;
   else
      A1(n_bits+1)=0;
   end
   if mod(length(ones_A2),2)==0
      A2(n_bits+1)=1;
   else
      A2(n_bits+1)=0;
   end
```

```
if mod(length(ones_B1),2)==0
   B1(n_bits+1)=1;
else
   B1(n_bits+1)=0;
end
if mod(length(ones_B2),2)==0
   B2(n_bits+1)=1;
else
   B2(n_bits+1)=0;
end

fprintf('\nA1:\t')
fprintf('%d', A1)
fprintf('\nA2:\t')
fprintf('%d', A2)
if mitm == 'Y'
   fprintf('\n--------------------\t')
   fprintf('\nE:\t')
   fprintf('%d', E)
end
fprintf('\n--------------------\t')
fprintf('\nB1:\t')
fprintf('%d', B1)
fprintf('\nB2:\t')
fprintf('%d', B2)

% Reconciled key: rk_A, rk_B
k_size=0; i=0;
while i<n_bits
   if A1(n_bits-i) == B1(n_bits-i)
      k_size = k_size+1;
      i=i+1;
   else
      i=i+1;
   end
end
if k_size<=1
   fprintf('\n')
   fprintf('\nInvalid reconciled key')
   return;
end
rk_A = rand(1, k_size) > 0.5;
```

```
rk_B = rand(1, k_size) > 0.5;
n=0; i=0;
while i<n_bits
   if A1(n_bits-i) == B1(n_bits-i)
      rk_A(k_size-n) = A2(n_bits-i);
      rk_B(k_size-n) = B2(n_bits-i);
      n=n+1;
      i=i+1;
   else
      i=i+1;
   end
end
n_error=0; i=0;
while i<k_size
   if rk_A(k_size-i) == rk_B(k_size-i)
      i=i+1;
   else
      n_error = n_error+1;
      i=i+1;
   end
end
BER_rk = (n_error/k_size)*100;

fprintf('\n')
fprintf('\nReconciled Key')
fprintf('\nAlice:\t')
fprintf('%d', rk_A)
fprintf('\nBob:\t')
fprintf('%d', rk_B)
fprintf('\nBER:\t')
fprintf('%.2f%%', BER_rk)

% Secret key: sk_A, sk_B
sk_A = rand(1, n_bits) > 0.5;
sk_B = rand(1, n_bits) > 0.5;
i=0;
while i<k_size
   sk_A(k_size-i) = rk_A(k_size-i);
   sk_B(k_size-i) = rk_B(k_size-i);
   i=i+1;
end
i=1;
```

```
    while i<=(n_bits-k_size)
        sk_A(k_size+i) = mod(sk_A(i)+sk_A(i+1),2);
        sk_B(k_size+i) = mod(sk_B(i)+sk_B(i+1),2);
        i=i+1;
    end
    n_error=0; i=0;
    while i<n_bits
        if sk_A(n_bits-i) == sk_B(n_bits-i)
            i=i+1;
        else
            n_error = n_error+1;
            i=i+1;
        end
    end
    BER_sk = (n_error/n_bits)*100;

    fprintf('\n')
    fprintf('\nSecret Key')
    fprintf('\nAlice:\t')
    fprintf('%d', sk_A)
    fprintf('\nBob:\t')
    fprintf('%d', sk_B)
    fprintf('\nBER:\t')
    fprintf('%.2f%%', BER_sk)

    % Eficiencia
    eff = ((k_size-n_error)/n_bits)*100;

    fprintf('\n')
    fprintf('\nProtocol Efficiency')
    fprintf('\nEff:\t')
    fprintf('%.2f%%\n', eff)

% Protocolo B92
elseif n_protocol == 2
    fprintf('\nB92 PROTOCOL\n')

    n_bytes = input('\nInput bytes: ');
    n_bits = n_bytes*8;
    if n_bits<=0
        fprintf('\nThe input bytes must be real number greater than zero')
        return;
```

```
    end
    prompt = ('\nMan in the middle [Y/N]: ');
    mitm = input(prompt, 's');
    if mitm == 'Y'
        fprintf('\nALICE ----- (Channel 1) ----- EVE ----- (Channel 2) ----- BOB\n');
        prob_channel1 = input('Error prob. in 1st channel [%]: ');
        prob_channel1 = prob_channel1/100;
        prob_measure1 = input('Error prob. in Eve measure [%]: ');
        prob_measure1 = prob_measure1/100;
        prob_channel2 = input('Error prob. in 2nd channel [%]: ');
        prob_channel2 = prob_channel2/100;
        prob_measure2 = input('Error prob. in Bob measure [%]: ');
        prob_measure2 = prob_measure2/100;
         if prob_channel1<0 || prob_channel1>1 || prob_channel2<0 || prob_channel2>1
|| prob_measure1<0 || prob_measure1>1 || prob_measure2<0 || prob_measure2>1
                fprintf('\nThe probability must be real number between zero and one
hundred')
            return;
        end
    elseif mitm == 'N'
        prob_measure1 = 0;
        prob_channel2 = 0;
        fprintf('\nALICE ----- (Channel) ----- BOB\n');
        prob_channel1 = input('Error prob. in the channel [%]: ');
        prob_channel1 = prob_channel1/100;
        prob_measure2 = input('Error prob. in Bob measure [%]: ');
        prob_measure2 = prob_measure2/100;
                    if prob_channel1<0 || prob_channel1>1 || prob_measure2<0 ||
prob_measure2>1
                fprintf('\nThe probability must be real number between zero and one
hundred')
            return;
        end
    else
        fprintf('\nThe input letter must be Y or N')
        return;
    end

    % Alice genera un vector de bits clasicos aleatorio: A
    A = rand(1, n_bits) > 0.5;
```

```
% Bob genera un vector de bits clasicos aleatorio: B1
B1 = rand(1, n_bits) > 0.5;

% Eva genera un vector de bits clasicos aleatorio: E
E = rand(1, n_bits) > 0.5;

% Alice genera un vector de bits clasicos resultante: A_res
A_res = rand(1, n_bits) > 0.5;
i=0;
while i<n_bits
   if A(n_bits-i)==0
      A_res(n_bits-i)=0;% Horizontal - |0>
      i=i+1;
   else
      A_res(n_bits-i)=1;% Right - |+>
      i=i+1;
   end
end

% Error de transmision de datos: Canal 1 (Alice - Eva)
i=0;
while i<n_bits
   prob_rand = 0 + (1-0).*rand(1,1);
   if prob_rand <= prob_channel1
      A_res(n_bits-i) = randi([0,1]);
      i=i+1;
   else
      i=i+1;
   end
end

% Eva genera un vector de bits clasicos resultante: E_res
E_res = rand(1, n_bits);
if mitm == 'Y'
   i=0;
   while i<n_bits
      prob_rand = 0 + (1-0).*rand(1,1);
      if E(n_bits-i) == A_res(n_bits-i)
         if prob_rand >= prob_measure1
            E_res(n_bits-i)=0;
         end
         i=i+1;
```

```
 else
          E_res(n_bits-i) = randi([0,1]);
          i=i+1;
       end
   end
end

% Error de medida de datos: Medida 1 (Eva)
i=0;
while i<n_bits
   prob_rand = 0 + (1-0).*rand(1,1);
   if prob_rand <= prob_measure1
      E_res(n_bits-i) = randi([0,1]);
      i=i+1;
   else
      if mitm == 'Y'
         i=i+1;
      else
         E_res(n_bits-i) = A_res(n_bits-i);
         i=i+1;
      end
   end
end

% Error de transmision de datos: Canal 2 (Eva - Bob)
i=0;
while i<n_bits
   prob_rand = 0 + (1-0).*rand(1,1);
   if prob_rand <= prob_channel2
      E_res(n_bits-i) = randi([0,1]);
      i=i+1;
   else
      i=i+1;
   end
end

% Bob genera un vector de bits clasicos: B2
B2 = rand(1, n_bits) > 0.5;
i=0;
while i<n_bits
   prob_rand = 0 + (1-0).*rand(1,1);
   if B1(n_bits-i) == E_res(n_bits-i)
```

```
        % Error de medida de datos: Medida 2 (Bob)
        if prob_rand >= prob_measure2
            B2(n_bits-i)=0;
        end
        i=i+1;
    else
        i=i+1;
    end
end

% Deteccion de errores: Bit de paridad
ones_A = find(A==1);
ones_B1 = find(B1==1); ones_B2 = find(B2==1);
if mod(length(ones_A),2)==0
    A(n_bits+1)=1;
else
    A(n_bits+1)=0;
end
if mod(length(ones_B1),2)==0
    B1(n_bits+1)=1;
else
    B1(n_bits+1)=0;
end
if mod(length(ones_B2),2)==0
    B2(n_bits+1)=1;
else
    B2(n_bits+1)=0;
end

fprintf('\nA:\t')
fprintf('%d', A)
if mitm == 'Y'
    fprintf('\n--------------------\t')
    fprintf('\nE:\t')
    fprintf('%d', E)
end
fprintf('\n--------------------\t')
fprintf('\nB1:\t')
fprintf('%d', B1)
fprintf('\nB2:\t')
fprintf('%d', B2)
```

```
% Reconciled key: rk_A, rk_B
k_size=0; i=0;
while i<n_bits
   if B2(n_bits-i)==1
      k_size = k_size+1;
      i=i+1;
   else
      i=i+1;
   end
end
if k_size<=1
   fprintf('\n')
   fprintf('\nInvalid reconciled key')
   return;
end
rk_A = rand(1, k_size) > 0.5;
rk_B = rand(1, k_size) > 0.5;
n=0; i=0;
while i<n_bits
   if B2(n_bits-i)==1
      rk_A(k_size-n) = A(n_bits-i);
      rk_B(k_size-n) = B1(n_bits-i);
      n=n+1;
      i=i+1;
   else
      i=i+1;
   end
end
i=0;
while i<k_size
   if rk_B(k_size-i)==1
      rk_B(k_size-i)=0;
      i=i+1;
   else
      rk_B(k_size-i)=1;
      i=i+1;
   end
end
n_error=0; i=0;
while i<k_size
   if rk_A(k_size-i) == rk_B(k_size-i)
      i=i+1;
```

```
    else
        n_error = n_error+1;
        i=i+1;
    end
end
BER_rk = (n_error/k_size)*100;

fprintf('\n')
fprintf('\nReconciled Key')
fprintf('\nAlice:\t')
fprintf('%d', rk_A)
fprintf('\nBob:\t')
fprintf('%d', rk_B)
fprintf('\nBER:\t')
fprintf('%.2f%%', BER_rk)

% Secret key: sk_A, sk_B
sk_A = rand(1, n_bits) > 0.5;
sk_B = rand(1, n_bits) > 0.5;
i=0;
while i<k_size
    sk_A(k_size-i) = rk_A(k_size-i);
    sk_B(k_size-i) = rk_B(k_size-i);
    i=i+1;
end
i=1;
while i<=(n_bits-k_size)
    sk_A(k_size+i) = mod(sk_A(i)+sk_A(i+1),2);
    sk_B(k_size+i) = mod(sk_B(i)+sk_B(i+1),2);
    i=i+1;
end
n_error=0; i=0;
while i<n_bits
    if sk_A(n_bits-i) == sk_B(n_bits-i)
        i=i+1;
    else
        n_error = n_error+1;
        i=i+1;
    end
end
BER_sk = (n_error/n_bits)*100;
```

```
    fprintf('\n')
    fprintf('\nSecret Key')
    fprintf('\nAlice:\t')
    fprintf('%d', sk_A)
    fprintf('\nBob:\t')
    fprintf('%d', sk_B)
    fprintf('\nBER:\t')
    fprintf('%.2f%%', BER_sk)

    % Eficiencia
    eff = ((k_size-n_error)/n_bits)*100;

    fprintf('\n')
    fprintf('\nProtocol Efficiency')
    fprintf('\nEff:\t')
    fprintf('%.2f%%\n', eff)

% Protocolo EPR
elseif n_protocol == 3
    fprintf('\nEPR PROTOCOL\n')

    n_bytes = input('\nInput bytes: ');
    n_bits = n_bytes*8;
    if n_bits<=0
        fprintf('\nThe input bytes must be real number greater than zero')
        return;
    end
    prompt = ('\nMan in the middle [Y/N]: ');
    mitm = input(prompt, 's');
    if mitm == 'Y'
        fprintf('\nALICE ----- (Channel 1) ----- EVE ----- (Channel 2) ----- BOB\n');
        prob_channel1 = input('Error prob. in 1st channel [%]: ');
        prob_channel1 = prob_channel1/100;
        prob_measure1 = input('Error prob. in Eve measure [%]: ');
        prob_measure1 = prob_measure1/100;
        prob_channel2 = input('Error prob. in 2nd channel [%]: ');
        prob_channel2 = prob_channel2/100;
        prob_measure2 = input('Error prob. in Bob measure [%]: ');
        prob_measure2 = prob_measure2/100;
                        if prob_channel1<0 || prob_channel1>1 || prob_channel2<0 ||
prob_channel2>1 || prob_measure1<0 || prob_measure1>1 || prob_measure2<0 ||
prob_measure2>1
```

```
                fprintf('\nThe probability must be real number between zero and one
hundred')
            return;
        end
    elseif mitm == 'N'
        prob_measure1 = 0;
        prob_channel2 = 0;
        fprintf('\nALICE ----- (Channel) ----- BOB\n');
        prob_channel1 = input('Error prob. in the channel [%]: ');
        prob_channel1 = prob_channel1/100;
        prob_measure2 = input('Error prob. in Bob measure [%]: ');
        prob_measure2 = prob_measure2/100;
                    if prob_channel1<0 || prob_channel1>1 || prob_measure2<0 ||
prob_measure2>1
                fprintf('\nThe probability must be real number between zero and one
hundred')
            return;
        end
    else
        fprintf('\nThe input letter must be Y or N')
        return;
    end

    % Alice genera un vector de bits clasicos aleatorio: A
    A = rand(1, n_bits) > 0.5;

    % Bob genera un vector de bits clasicos aleatorio: B1
    B1 = rand(1, n_bits) > 0.5;

    % Eva genera un vector de bits clasicos aleatorio: E
    E = rand(1, n_bits) > 0.5;

    % Alice genera dos vectores de bits clasicos resultantes: A1_res, A2_res
    A1_res = rand(1, n_bits); A2_res = rand(1, n_bits);
    A_res = rand(1, n_bits); A_binres = rand(1, n_bits) > 0.5;
    i=0;
    while i<n_bits
        if A(n_bits-i)==0
            A1_res(n_bits-i)=0;% Horizontal - |0>
            A2_res(n_bits-i)=1;% Vertical - |1>
            i=i+1;
        else
```

```
A1_res(n_bits-i)=2;% Right - |+>
A2_res(n_bits-i)=3;% Left - |->
        i=i+1;
    end
end
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if prob_rand >= 0.5
        A_res(n_bits-i) = A1_res(n_bits-i);
        i=i+1;
    else
        A_res(n_bits-i) = A2_res(n_bits-i);
        i=i+1;
    end
end
i=0;
while i<n_bits
    if A_res(n_bits-i)==0 || A_res(n_bits-i)==2
        A_binres(n_bits-i)=0;
        i=i+1;
    else
        A_binres(n_bits-i)=1;
        i=i+1;
    end
end

% Error de transmision de datos: Canal 1 (Alice - Eva)
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if prob_rand <= prob_channel1
        A_res(n_bits-i) = randi([0,3]);
        i=i+1;
    else
        i=i+1;
    end
end

% Eva genera un vector de bits clasicos resultante: E_res
E_res = rand(1, n_bits);
if mitm == 'Y'
```

```
    i=0;
    while i<n_bits
        prob_rand = 0 + (1-0).*rand(1,1);
        if E(n_bits-i)==0 && (A_res(n_bits-i)==0 || A_res(n_bits-i)==1)
            if A_res(n_bits-i)==0
                if prob_rand >= prob_measure1
                    E_res(n_bits-i)=0;% Horizontal - |0>
                end
            else
                if prob_rand >= prob_measure1
                    E_res(n_bits-i)=1;% Vertical - |1>
                end
            end
            i=i+1;
        elseif E(n_bits-i)==1 && (A_res(n_bits-i)==2 || A_res(n_bits-i)==3)
            if A_res(n_bits-i)==2
                if prob_rand >= prob_measure1
                    E_res(n_bits-i)=2;% Right - |+>
                end
            else
                if prob_rand >= prob_measure1
                    E_res(n_bits-i)=3;% Left - |->
                end
            end
            i=i+1;
        else
            E_res(n_bits-i) = randi([0,3]);
            i=i+1;
        end
    end
end

% Error de medida de datos: Medida 1 (Eva)
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if prob_rand <= prob_measure1
        E_res(n_bits-i) = randi([0,3]);
        i=i+1;
    else
        if mitm == 'Y'
            i=i+1;
```

```
        else
            E_res(n_bits-i) = A_res(n_bits-i);
            i=i+1;
        end
    end
end

% Error de transmision de datos: Canal 2 (Eva - Bob)
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if prob_rand <= prob_channel2
        E_res(n_bits-i) = randi([0,3]);
        i=i+1;
    else
        i=i+1;
    end
end

% Bob genera un vector de bits clasicos: B2
B2 = rand(1, n_bits) > 0.5;
i=0;
while i<n_bits
    prob_rand = 0 + (1-0).*rand(1,1);
    if B1(n_bits-i)==0 && (E_res(n_bits-i)==0 || E_res(n_bits-i)==1)
        if E_res(n_bits-i)==0
            % Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
                B2(n_bits-i)=0;
            end
        else
            % Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
                B2(n_bits-i)=1;
            end
        end
        i=i+1;
    elseif B1(n_bits-i)==1 && (E_res(n_bits-i)==2 || E_res(n_bits-i)==3)
        if E_res(n_bits-i)==2
            % Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
                B2(n_bits-i)=0;
```

```
            end
        else
            % Error de medida de datos: Medida 2 (Bob)
            if prob_rand >= prob_measure2
                B2(n_bits-i)=1;
            end
        end
        i=i+1;
    else
        i=i+1;
    end
end

% Deteccion de errores: Bit de paridad
ones_A = find(A==1);
ones_B1 = find(B1==1); ones_B2 = find(B2==1);
if mod(length(ones_A),2)==0
    A(n_bits+1)=1;
else
    A(n_bits+1)=0;
end
if mod(length(ones_B1),2)==0
    B1(n_bits+1)=1;
else
    B1(n_bits+1)=0;
end
if mod(length(ones_B2),2)==0
    B2(n_bits+1)=1;
else
    B2(n_bits+1)=0;
end

fprintf('\nA:\t')
fprintf('%d', A)
if mitm == 'Y'
    fprintf('\n--------------------\t')
    fprintf('\nE:\t')
    fprintf('%d', E)
end
fprintf('\n--------------------\t')
fprintf('\nB1:\t')
fprintf('%d', B1)
```

```
fprintf('\nB2:\t')
fprintf('%d', B2)

% Reconciled key: rk_A, rk_B
k_size=0; i=0;
while i<n_bits
   if A(n_bits-i) == B1(n_bits-i)
      k_size = k_size+1;
      i=i+1;
   else
      i=i+1;
   end
end
if k_size<=1
   fprintf('\n')
   fprintf('\nInvalid reconciled key')
   return;
end
rk_A = rand(1, k_size);
rk_B = rand(1, k_size);
n=0; i=0;
while i<n_bits
   if A(n_bits-i) == B1(n_bits-i)
      rk_A(k_size-n) = A_binres(n_bits-i);
      rk_B(k_size-n) = B2(n_bits-i);
      n=n+1;
      i=i+1;
   else
      i=i+1;
   end
end
n_error=0; i=0;
while i<k_size
   if rk_A(k_size-i) == rk_B(k_size-i)
      i=i+1;
   else
      n_error = n_error+1;
      i=i+1;
   end
end
BER_rk = (n_error/k_size)*100;
```

```
fprintf('\n')
fprintf('\nReconciled Key')
fprintf('\nAlice:\t')
fprintf('%d', rk_A)
fprintf('\nBob:\t')
fprintf('%d', rk_B)
fprintf('\nBER:\t')
fprintf('%.2f%%', BER_rk)

% Secret key: sk_A, sk_B
sk_A = rand(1, n_bits) > 0.5;
sk_B = rand(1, n_bits) > 0.5;
i=0;
while i<k_size
   sk_A(k_size-i) = rk_A(k_size-i);
   sk_B(k_size-i) = rk_B(k_size-i);
   i=i+1;
end
i=1;
while i<=(n_bits-k_size)
   sk_A(k_size+i) = mod(sk_A(i)+sk_A(i+1),2);
   sk_B(k_size+i) = mod(sk_B(i)+sk_B(i+1),2);
   i=i+1;
end
n_error=0; i=0;
while i<n_bits
   if sk_A(n_bits-i) == sk_B(n_bits-i)
      i=i+1;
   else
      n_error = n_error+1;
      i=i+1;
   end
end
BER_sk = (n_error/n_bits)*100;

fprintf('\n')
fprintf('\nSecret Key')
fprintf('\nAlice:\t')
fprintf('%d', sk_A)
fprintf('\nBob:\t')
fprintf('%d', sk_B)
fprintf('\nBER:\t')
```

```
    fprintf('%.2f%%', BER_sk)

    % Eficiencia
    eff = ((k_size-n_error)/n_bits)*100;

    fprintf('\n')
    fprintf('\nProtocol Efficiency')
    fprintf('\nEff:\t')
    fprintf('%.2f%%\n', eff)

else
    fprintf('\n')
    fprintf('\nThe input must be real number between 1 and 3')
end
```

A.5. Código Matlab "Quantum channel"

QC.m

```
fprintf('\nQUANTUM CHANNEL SIMULATOR\n')
fprintf('\n1.Optical Fiber   2.Free Space')
n_channel = input('\nChoose a quantum channel number: ');

% Canal: Fibra Optica
if n_channel == 1
    fprintf('\nOptical Fiber\n')
    d = input('\nInput distance [km]: ');
    alfa = input('\nInput attenuation [0.2 or 0.34 dB/km]: ');% Coeficiente de
perdidas por la distancia [dB/km]

    % Modelo simple de perdidas para fibra optica
    l = 10^(-(alfa*d)/10);
    l_dB = -10*log10(l);

    fprintf('\nChannel losses [dB]:\t');
    fprintf('%.2f%\n', l_dB);

% Canal: Espacio libre
elseif n_channel == 2
    fprintf('\nFree Space Optical Channel\n')
    d = input('\nInput distance [km]: ');
    fprintf('\n1.Clear  2.Rain  3.Snow  4.Fog')
    meteo = input('\nChoose a meteorological phenomena number: ');
    if meteo == 1
```

```
 alfa = 0.1;% Coeficiente de perdidas por la distancia [dB/km]
  elseif meteo == 2
     alfa = 1;
  elseif meteo == 3
     alfa = 3;
  elseif meteo == 4
     alfa = 10;
  else
     fprintf('\n')
     fprintf('\nThe input must be real number between 1 and 2')
  end
  dt = 3*10^(-2);% Apertura del emisor [m]
  dr = 8*10^(-2);% Apertura del receptor [m]
  D = 2; % Divergencia del haz [mrad]

  % Modelo simple de perdidas para espacio libre
  l = 10^(-(alfa*d)/10);
  lgeo = (dr/(dt+D*d))^2;% Perdidas geometricas
  ltot_dB = -(10*log10(l)+10*log10(lgeo));

  fprintf('\nChannel losses [dB]:\t');
  fprintf('%.2f%\n', ltot_dB);

else
  fprintf('\n')
  fprintf('\nThe input must be real number between 1 and 2')
end
```